Carmen MINCĂ *(coord.)*

Alina Gabriela BOCA
Radu BORIGA
Luminița CIOCARU
Georgeta CRĂCIUNESCU
Elena DRĂGAN
Doina DRUȚĂ
Nușa DUMITRIU-LUPAN
Diana GHINEA
Maria NIȚĂ

Rodica PINTEA *(coord.)*

Adrian NIȚĂ
Oana-Maria LUPAȘCU
Victor MANZ
Marius MÎRZACU
Ștefania PENEA
Luminița RADU
Silviu SĂCUIU
Bogdan SICHIM
Corina Elena VINȚ

BACALAUREAT

la Informatică

TESTE REZOLVATE - 2021

L&S SOFT
INFOBITS.RO

Descrierea CIP a Bibliotecii Naţionale a României
Bacalaureat 2021 la informatică, teste rezolvate. - Voluntari :
Editura L & S Soft Management Impex S.R.L., 2021.
ISBN 978-606-94898-5-7

004

Vă recomandăm ...



Limbajul **Python 3** este folosit la scară largă în prezent, fiind foarte popular, simplu de învățat și de utilizat. Spre deosebire de alte limbaje de programare, **Python 3** se înțelege foarte repede, *chiar de la început*.

Curs online gratuit. Studiază lecțiile și elementele de teorie, analizează problemele rezolvate și propuse, efectuează testele grilă (*atenție la note*) ... și obții ***Diploma*** pentru ***Programator Junior***:

www.pythonisti.ro



Editura L&S SOFT

Adresa: Aleea Aviatiei nr. 10, Voluntari, Ilfov;
Mobil: 0727.731.947;
E-mail: hello@infobits.ro;

www.ls-infomat.ro
www.manuale-de-informatica.ro

Infobits Academy
www.infobits.ro

CUPRINS

VARIANTA 1

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Indicaţi expresia care are valoarea **`true`/1** dacă şi numai dacă numărul real memorat în variabila **x** se află în intervalul `[-5,5]`.

	Limbajul Pascal	Limbajul C/C++
a.	`x*x-25<0`	`x*x-25<0`
b.	`25-x*x>=0`	`25-x*x>=0`
c.	`(5<=x)and(x<=-5)`	`(5<=x)&&(x<=-5)`
d.	`(x-5)*(x+5)>=0`	`(x-5)*(x+5)>=0`

2. Pentru subprogramul **`ex`** cu definiția următoare, indicați ce se va afișa în urma apelului **ex(25436)**.

Limbajul Pascal

```
procedure ex( x:longint);
begin
  if x>3 then begin
       write(x mod 10);
       ex(x div 10) end
  else write('*')
end;
```

Limbajul C/C++

```
void ex(int x)
{ if(x>3) {
    printf("%d",x%10); | cout<<x%10;
    ex(x/10);
  }
  else printf("*"); | cout<<"*";
}
```

a. 6345*	b. 2345*	c.65432*	d. *6543

3. Utilizând metoda backtracking se generează în ordine crescătoare toate numerele naturale de câte patru cifre din mulţimea `A={1,2,3,4,5}`, numere care nu conţin pe poziții alăturate cifre din mulţimea `{1,5}`.Primele opt numere generate sunt, în ordine: `1212`, `1213`, `1214`, `1221`, `1222`, `1223`, `1224`, `1225`. Câte dintre numerele generate încep cu cifra `2` şi se termină cu cifra `5`?

a. 9	b. 12	c. 15	d. 19

4. Câte grafuri neorientate, distincte, cu `5` vârfuri se pot construi? Două grafuri se consideră distincte dacă matricele lor de adiacenţă sunt diferite.

a. 5	b. 120	c. 2^{10}	d. 4^{10}

5. Stabiliți care dintre următoarele grafuri cu `6` noduri, numerotate de la `1` la `6,` date prin listele de adiacenţă reprezintă un graf neorientat conex și aciclic.

a.	b.	c.	d.
1:2,4; 2:1,3; 3:2,4,5,6; 4:1,3; 5:3,6; 6:3,5;	1:2,3; 2:1,3; 3:1,2; 4:5; 5:4,6: 6:5:	1:2,3; 2:1,3; 3:1,2; 4:; 5:6 6:5	1:2,6; 2:1,3; 3:2,4; 4:3,5; 5:4,6; 6:1,5;

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citește n (număr natural)
z←0; p←1
┌cât timp n>0 execută
│ c←n%10; n←[n/10]
│┌dacă c%2=0 atunci
││ z←z+p*(c+1); p←p*10
│└■
└■
scrie z
```

a) Ce valoare va fi afişată dacă se citeşte valoarea **1092744**? **(6p.)**

b) Scrieţi toate numere naturale, distincte, fiecare având **exact** două cifre, care pot fi citite pentru variabila **n** astfel încât să se afişeze valoarea **7**. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **cât timp...execută** cu o structură repetitivă de alt tip.**(6p.)**

d) Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. În declararea următoare, câmpurile **x** şi **y** ale înregistrării pot memora coordonatele carteziene ale unui punct din planul **xOy**.

Limbajul Pascal	Limbajul C/C++
`type punct=record x,y:real end;` `var A,B:punct;`	`struct punct { float x,y;} A,B;`

Scrieţi o expresie în limbajul **Pascal** | **C/C++** care să aibă valoarea **true**|1 dacă şi numai dacă punctele memorate în variabilele **A** şi **B** sunt situate la egală distanţă faţă de originea axelor de coordonate, **O**. **(6p.)**

3. În secvenţa de program alăturată, variabila **s** memorează un şir cu cel mult 30 de caractere, iar variabila **i** este de tip întreg. Completaţi punctele de suspensie din secvenţă astfel încât, în urma executării secvenţei, aceasta să afişeze şirul de caractere **bac2021cunota10**. **(6p.)**

Limbajul Pascal

```
s:='bac2723cunota37';
for i:=1 to length(s) do
  if .... then write(....)
  else if.... then write(....)
         else  write(....);
```

Limbajul C/C++

```
strcpy(s,"bac2723cunota37");
for(i=0;i<strlen(s);i++)
if(....)cout<<....;
  else if(.....)cout<<...;
       else cout<< ....;
```

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Scrieţi definiţia completă a unui subprogram **sub**, cu patru parametri, care primeşte prin intermediul parametrilor:

- **a** şi **b**, două cifre distincte **(a<10, b<10, a≠b);**
- **n**, un număr natural nenul (0 < **n** < 100);
- **v**, un tablou unidimensional care memorează un şir de **n** numere naturale, fiecare având cel mult nouă cifre.

Subprogramul **sub** determină modificarea şirului de numere primit prin parametrul **v** realizând eliminarea tuturor numerelor care conţin în scrierea lor cel puţin o cifră **a** şi nu conțin nicio cifră **b**. De asemenea, subprogramul va furniza prin parametrului **n** numărul de numere din şirul modificat, iar prin parametrul **v** tabloul modificat. Dacă în şirul primit nu există niciun număr care să conţină în scrierea sa cifra **a** și să nu conțină cifra **b**, atunci valorile parametrilor **n** și **v** nu se vor modifica.

Exemplu. Pentru **n=10, v=(3551,149,3798,502,75,2515, 51, 151, 489, 653)**, **a=5** şi **b=2** ale parametrilor, în urma apelului, subprogramului **sub** va furniza prin parametrii indicaţi valorile **n=5** şi **v=(149, 3798, 502, 2515, 489). (10p.)**

2. Să se construiască o matrice cu **n** linii şi **m** coloane care conţine pe prima coloană, de sus în jos, toate numerele naturale de la **1** la **n**, în ordine crescătoare, pe coloana a doua, de sus în jos, toate numerele naturale de la **2** la **n+1**, în ordine crescătoare, pe a treia coloană, de sus în jos, toate numerele naturale de la **3** la **n+2**, în ordine crescătoare, pe a patra coloană, de sus în jos, toate numerele naturale de la **4** la **n+3**, în ordine crescătoare şi aşa mai departe până la coloana m. Scrieţi programul care citeşte de la tastatură două valori naturale n și m (**2<n<25**, **2<m<25**), construieşte matricea conform cerinţei şi o afişează pe ecran, pe linii, cu spaţii între elementele de pe fiecare linie.

De exemplu, dacă se citeşte de la tastatură **n=3** și **m=5**, se va construi şi se va afişa pe ecran matricea:

1 2 3 4 5
2 3 4 5 6
3 4 5 6 7 **(10p.)**

3. Fişierul text bac.in conţine un şir **s** de cel mult un milion de naturale nenule, formate fiecare din cel mult **9** cifre, separate prin câte un spaţiu.

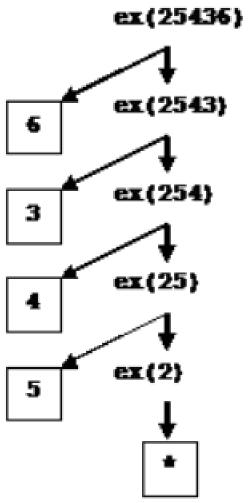
a) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină și afișează pe ecran lungimea maximă a unei secvenţe din şir formată doar din numere pare. O secvenţă a unui şir constă în elemente aflate pe poziţii consecutive în şirul considerat. **(8p.)**

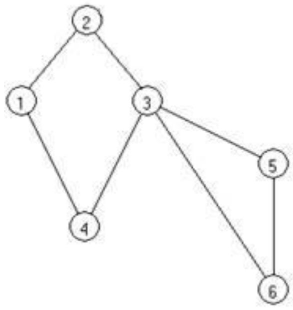
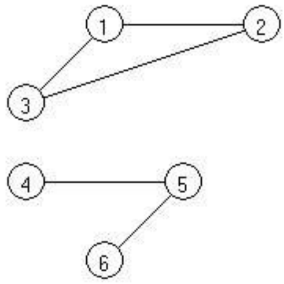
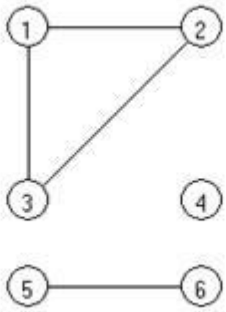
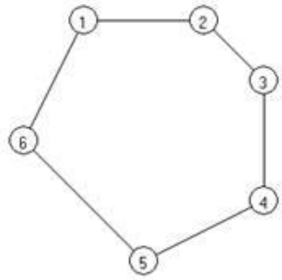
Exemplu: dacă fişierul **bac.in** are conţinutul:
12 6 245 18 8 2 36 39 34 8 36 11 10 12 102 24 881
atunci, pe ecran se va afişa numărul **4** reprezentând lungimea maximă a unei secvenţe formată doar din numere pare din şirul dat.

b) Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

VARIANTA 1 - *Rezolvare*

Subiectul 1

1.	`25-x*x>=0 <> x*x-25<=0 <> (x-5)*(x+5)<=0 <> x∈[-5,5].` Pentru precizarea răspunsului **b)** se acordă **4p**.
2.	În desenul următor este prezentat mecanismul prin care se execută apelul **`ex(25436)`** al subprogramului recursiv **`ex`**. Se vor afişa, în ordine: **`6345*.`** Pentru răspunsul a**)** se acordă **4p**. 
3.	Numerele cerute sunt: **`2125; 2135; 2145; 2225; 2235; 2245; 2325; 2335; 2345; 2425;2435; 2445; 2525; 2535; 2545`**. Sunt generate **`15`** numere cu proprietatea din enunţ. Pentru răspunsul **c)** se acordă **4p**.
4.	Fiecărui graf îi corespunde o funcţie `f:{(i,j)|i≠j, 1≤i,j≤5}→{0,1}` care asociază perechii de noduri distincte **`(i,j)`** valoarea: • **`0`** dacă nu există muchie de la **`i`** la **`j`** • **`1`** dacă există muchie de la **`i`** la **`j`**. Numărul grafurilor cu câte **`n`** vârfuri este egal cu numărul acestor funcţii, adică $C_n^2 = 2^{\frac{n(n-1)}{2}}$. Pentru răspunsul c**)** se acordă **4p.**
5.	Pentru precizarea răspunsului **d)** se acordă **4p**.

	 a) Graf conex care conține cicluri (eulerian, nu este hamiltonian)	 b) Graf neconex
	 c) Graf neconex	 d) Graf conex aciclic (eulerian și hamiltonian)

Subiectul 2

1.	**a)**	Din numărul citit sunt eliminate toate cifrele impare iar fiecare cifră pară din număr este mărită cu `1`. Pentru precizarea valorii **`1355`** se acordă **6p**.
	b)	Valoarea 7 poate fi obținută din numere citite pentru n de forma **`6x`** sau **`x6`** cu **`x`**є{1,3,5,7,9} cifră impară. Algoritmul va afișa valoarea 7 dacă se citește pentru n orice valoare din mulțimea {**`16, 36, 56, 76, 96, 61, 63, 65, 67, 69`**}. Pentru precizarea valorilor **`16,36,56,76,96,61,63,65,67,69`** se acordă 6**p**. Pentru identificarea valorilor corecte cu adăugarea sau repetarea unora dintre valori se acordă 5p. Pentru identificarea a cel puțin 5 valori corecte se acordă 3p., iar pentru identificarea a cel puțin două valori corecte se acordă 1p. În celelalte cazuri se acordă 0p.
	c)	Înlocuirea structurii **`cât timp n>0 execută...`** presupune utilizarea unei structuri repetitive **`repetă`**...**`..cât timp`** sau **`repetă`**...**`.până când`**:

```
┌dacă n>0 atunci
│ ┌repetă
│ │  c←n%10; n←[n/10]
│ │ ┌dacă c%3=0 atunci
│ │ │  z←z+p*(9-c)
│ │ │  p←p*10
│ │ └■
│ └cât timp n>0
└■
```

```
┌dacă n>0 atunci
│ ┌repetă
│ │  c←n%10; n←[n/10]
│ │ ┌dacă c%3=0 atunci
│ │ │  z←z+p*(9-c)
│ │ │  p←p*10
│ │ └■
│ └până când n=0
└■
```

Fără testarea inițială a valorii lui **n** (**n>0**), algoritmul obținut prin înlocuirea mecanică a structurii **cât_timp** cu structura **repetă...** va afișa pentru **n=0** valoarea **1** spre deosebire de algoritmul din enunț care afișează valoarea 0 pentru **n=0**, deci cei doi algoritmi nu sunt echivalenți.

Pentru alegerea unei structuri repetitive se acordă 2p., pentru testarea inițială a valorii lui **n** se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 1p, iar pentru scrierea integrală a restului algoritmului se mai acordă 1p. - în total **6p**.

d) Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc.

Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afișare a rezultatului și de decizie se acordă câte 1p., pentru instrucțiunea repetitivă se acordă 3p., iar pentru cele 6 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.

2. **Un răspuns corect:**

```
Pascal: A.x*A.x+A.y*A.y = B.x*B.x+B.y*B.y
C/C++:  A.x*A.x+A.y*A.y == B.x*B.x+B.y*B.y
```

Pentru orice răspuns corect se acordă **6p**.

3. Un răspuns corect:

Limbajul Pascal

```
s:='bac2723denota37';
for i:=1 to length(s) do
  if s[i]='7' then write(0)
  else if s[i]='3' then write(1)
       else  write(s[i]);
```

Limbajul C/C++

```
strcpy(s,"bac2723denota37");
for(i=0;i<strlen(s);i++)
 if(s[i]=='7')cout<<0;
 else if(s[i]=='3')cout<<1;
      else cout<<s[i];
```

Subiectul 3

1. Prelucrarea oferită de subprogram verifică existența celor două cifre în scrierea fiecărui număr memorat în tablou. Ștergerea unui element care conține cifra **a** și nu conține cifra **b** se realizează prin deplasarea cu o poziție la stânga a tuturor elementelor situate la dreapta celui ce urmează a fi șters. După realizarea deplasării, se micșorează valoarea lui **n** cu **1**.

Limbajul Pascal

```
type vector=array[1..100]of longint;
procedure sub(a:longint; b:byte;
              var n:byte; var v:vector);
var exa,exb:boolean;
        uc,i,j:byte;
        z:longint;
begin
        i:=1;
        while i<=n do begin
                exa:=false; exb:=false; z:=v[i];
                while z>0 do
                  begin
                        uc:=z mod 10;
                        z:=z div 10;
                        if uc=a then exa:=true
                                else if uc=b then exb:=true;
                  end;
                if exa=true and exb=false then
                 begin
                        for j:=i+1 to n do v[j-1]:=v[j];
                        n:=n-1
                 end
                else inc(i);
        end
end;
```

Limbajul C/C++

```
void sub(int a, int b, int &n, int v[])
{
        int i=1,uc,exa,exb,z,j;
        while(i<=n)
        {
                exa=0; exb=0; z=v[i];
                while(z>0)
                {
                        uc=z%10;
                        z=z/10;
                        if (uc==a) exa=1;
                        else if(uc==b) exb=1;
                }
                if (exa && !exb)
                {
                        for(j=i+1;j<=n;j++) v[j-1]=v[j];
                        n--;
                }
                else i++;
        }
}
```

Pentru antet corect se acordă 3p. (Pentru respectarea structurii antetului (procedure/void) se acordă 1p., pentru declararea corectă a parametrilor de intare a și b încă 1p., pentru declararea corectă a parametrilor **n** și **v** încă 1p.).

Pentru declararea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p.

Pentru determinarea numărului de apariţii ale cifrei a în scrierea unui număr din tabloului 1p. Pentru determinarea numărului de apariţii ale cifrei **b** în scrierea unui număr din tabloului 1p. Pentru determinarea elementelor tabloului cu proprietatea cerută 1p. Pentru ștergerea din tablou a tuturor elementelor ce memorează numerele cu proprietatea cerută 1p. Pentru corectitudinea globală a programului 1p.

În total **10p.**

2. Pot fi concepute mai multe metode de obținere a matricei cerute. Metoda clasică ce completează coloanele matricei (numerotate de la **1** la **m**), respectiv liniile (numerotate de la **1** la **n**), în maniera descrisă, este:

```
┌pentru j←1,m execută
| k← j
|┌pentru i←1,n execută
|| A[i,j]←k
|| k← k+1
|└■
└■
```

Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui **n** și **m** se mai acordă 1p., pentru parcurgerea unei coloane sau linii se acordă 1p., pentru completarea corectă a unei coloane se acordă 2p., pentru completarea corectă a tuturor coloanelor se acordă 2p., iar pentru afișarea matricei conform cerinței se acordă 2p. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă 1p. - în total **10p**.

3. **b)** **Algoritmul de rezolvare propus** citește, unul câte unul, numerele din fișier, analizând paritatea fiecăruia: întâlnirea unui număr par va duce la "prelungirea" secvenței curente, iar întâlnirea unui număr impar va conduce la trecerea la o nouă secvență. La finele oricărei secvențe se analizează lungimea ei, în vederea memorării lungimii maxime.

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul. Deoarece numărul de valori din fișier (1000000) este dimensiunea care contează în problemă, spunem că am obținut un algoritm liniar. Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple.

Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. - în total **2p.**

a)

Limbajul Pascal

```
Var f:text;
     nr,lgmax,lg:longint;
```

```
begin
      assign(f,'bac.in');reset(f);
      while not eof(f) do begin
            read(f,nr);
            if nr mod 2=0 then
                  inc(lg)
            else begin
                  if lgmax<lg then
                        lgmax:=lg;
                  lg:=0
            end;
      end;
      if lgmax<lg then
            lgmax:=lg;
      close(f);
      write(lgmax)
end.
```

Limbajul C/C++

```
#include
#include
using namespace std;
ifstream f("bac.in");
int main()
{
        int lgmax,lg,nr;
        lg=lgmax=0;
        while(f>>nr)
                if(nr%2==0) lg++;
                else {
                        if(lgmax<lg) lgmax=lg;
                        lg=0;
                }
        if(lgmax<lg) lgmax=lg;
        cout<<lgmax;
        return 0;
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea valorii cerute, 1p. pentru afișarea ei, 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).
Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare ((O(n)) şi 1p. pentru utilizarea eficentă a memoriei (O(1)) . În total **8p.**

VARIANTA 2

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Știind că variabila a memorează un număr natural format din cel puțin două cifre, stabiliți care dintre expresiile **Pascal/C/C++** următoare are valoarea **true/1** dacă şi numai dacă valoarea variabilei `a` are cifra zecilor egală cifra unităților.

	Limbajul Pascal	Limbajul C/C++
a.	`a mod 10 div 10 = a div 10`	`a%10/10 == a/10`
b.	`a div 100 mod 10 = a mod 10`	`a/100%10 == a%10`
c.	`a div 10 mod 10 = a div 10`	`a/10%10 == a/10`
d.	`a div 10 mod 10 = a mod 10`	`a/10%10 == a%10`

2. Pentru subprogramul `f` cu definiția următoare, indicați cea mai mare valoare de **4** cifre pe care o poate avea o variabilă **n**, astfel încât pentru apelul **f(n)** să se obţină valoarea 1.

Limbajul Pascal

```
function f(x:integer) : integer;
begin
  if x=0 then  f:=1
  else
  if x mod 10<5 then f:=f(x div 10)
  else f:=x mod 10+f(x div 10)*10;
end;
```

Limbajul C/C++

```
int f(int x)
{
 if(x==0) return 1;
 else
 if(x%10<5) return f(x/10);
 else return x%10+f(x/10)*10;
}
```

a. 1431	b. 3034	c.4444	d. 9997

3. Utilizând metoda backtracking se generează în ordine alfabetică toate cuvintele formate din cel puțin trei litere din mulțimea `A={a,b,c,d,e}`, cuvinte care au literele distincte două câte două și nu conțin două consoane alăturate. Primele șase cuvinte generate sunt, în această ordine: **abe**, **abec**, **abed**, **ace**, **aceb**, **aced**. Indicați care sunt ultimele două cuvinte generate, în ordinea generării lor:

a. eda, edab	b. decab, edc	c. edac, edec	d. edab, edac

4. Se consideră arborele cu **10** noduri, etichetate cu numere distincte de la **1** a **10**, dat prin următorul vector de taţi: `(5,5,2,5,0,2,2,9,7,7)`. Ştiind că rădăcina se află pe nivelul 0, stabiliţi care este numărul ascendenţilor nodului **8** din arbore:

a.1	b. 2	c. 3	d. 4

5. Se consideră un graf neorientat G conex, cu **2021** noduri. Indicați care poate fi gradul minim și care poate fi gradul maxim al unui nod din graful G.

a. 0, 2020	b. 1,2020	c. 1,2021	d. 1,2020

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte a,b (numere naturale)
nr←0
┌cât timp a>0 şi b>0 execută
│ ┌dacă a %10 > b%10
│ │ atunci c←a%10; a←[a/10]
│ │ altfel c←b%10; b←[b/10]
│ └■
│ nr←nr*10+c
└■
scrie nr
```

a) Ce valoare va fi afişată dacă se citesc valorile **153** şi **204**? **(6p.)**

b) Știind că pentru variabila **b** s-a citit valoarea **1202**, scrieţi câte numere naturale (formate din exact patru cifre) pot fi citite pentru variabila **a** astfel încât, pentru fiecare astfel de valoare, să se afişeze valoarea **2021**. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **cât timp...execută** cu o structură repetitivă de alt tip. **(6p.)**

d) Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. Se consideră declarările următoare în care variabila **C** memorează informaţiile referitoare la un cerc din planul cu sistemul de coordonate **xOy**: coordonatele centrului cercului şi raza cercului în sistemul de coordonate **xOy**. Scrieţi o expresie în limbajul **Pascal|C/C++** care să aibă valoarea **true**/1 dacă şi numai dacă cercul (cu informaţiile memorate în variabila **C**) intersectează axa **Ox**. **(6p.)**

Limbajul Pascal	Limbajul C/C++
`Type punct=record` `x,y:integer end;` `cerc = record` `O:punct; R: integer;end;` `var C: cerc;`	`struct punct {` `int x,y; };` `struct cerc {` `struct punct O; int R; };` `cerc C;`

3. În secvenţa următoare, variabilele **i** şi **j** sunt de tip integer **inter/int**, iar variabila **A** memorează o matrice cu **6** linii şi **6** coloane (numerotate de la **0** la **5**) cu elemente de tip **integer/int**. Scrieți valorile elementelor memorate pe diagonala secundară a matricei **A** la finalul executării secvenţei, în ordinea crescătoare a indicilor liniilor pe care sunt situate. **(6p.)**

Limbajul Pascal

```
for i:= 0 to 5 do begin
   x:=i;
   for j:= 5 downto 0 do begin
               inc(x);
               A[i,j]:=x
         end
end
```

Limbajul C/C++

```
for(i=0;i<=5;i++) {

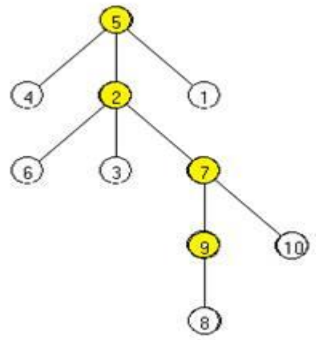
    x=i;
    for(j=5;j>=0;j--) {
                x++;
                A[i][j]=x;
        }
}
```

Subiectul 3 (30p)

	Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.
1.	Scrieţi în caseta text următoare definiţia completă a unui subprogram **sub**, cu doi parametri, care primeşte prin intermediul parametrilor: • **n**, un număr natural nenul (**0 < n < 100**); • **v**, un tablou unidimensional ce memorează un şir de **n** numere naturale, fiecare având cel mult patru cifre. Subprogramul **sub** determină cel mai mic și cel mai mare număr din şirul de numere primit prin parametrul **v** și modifică acest șir prin interschimbarea primei apariții a celui mai mic număr din șir cu ultima apariție a celui mai mare număr din șir. De asemenea subprogramul va furniza prin intermediul parametrului **v** tabloul modificat. Dacă șirul nu conține cel puțin două numere distincte, atunci șirul de numere primit prin parametrul **v** nu se va modifica. **Exemplu**. Pentru valorile **n=10**, **v=(35, 1, 52, 98, 1, 98, 51, 11, 98, 65)**, ale parametrilor, în urma apelului, subprogramului **sub** va furniza prin parametrul **v** tabloul: **(35, 98, 52, 98, 1, 98, 51, 11, 1, 65). (10p.)**
2.	Un şir cu maximum `255` de caractere conţine cuvinte formate numai din litere mici ale alfabetului englez. Fiecare cuvânt este urmat de un caracter `*`. Scrieţi un program **Pascal/C/C++** care citeşte un astfel de şir şi afişează pe ecran şirul obţinut prin eliminarea tuturor cuvintelor din şir care au ca sufix primul cuvânt din şir, ca în exemplu. Un cuvânt `a` este sufix pentru cuvântul `b` dacă există un cuvânt `c` de lungime mai mare sau egală cu `0`, astfel încât, prin alipirea cuvântului `a` la sfârşitul cuvântului `c`, se obţine cuvântul `b`. ***Exemplu***. Pentru şirul: `este*soarele*coboara*peste*creste*la*apus*amestecand*norii*` se va afişa: `*soarele*coboara***la*apus*amestecand*norii*` **(10p.)**
3.	Fişierul text bac.in conţine un şir **s** de cel mult un milion de numere naturale nenule, formate fiecare din cel mult **9** cifre, separate prin câte un spaţiu. **a**) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină și afișează pe ecran lungimea maximă a unei secvenţe din şir formată doar din numere care au suma cifrelor egală cu suma cifrelor primului număr din secvență. O secvenţă a unui şir constă în elemente situate pe poziţii consecutive în şirul considerat. **(8p.)** **Exemplu:** dacă fişierul **bac.in** are conţinutul: `1 100 10 245 13 22 4 39 5 32 401 3110 11 17 8 26 81` atunci, pe ecran se va afişa numărul **4** reprezentând lungimea maximă a unei secvenţe formată doar din numere care au fiecare suma cifrelor egală cu **5** și sunt situate pe poziții consecutive în şirul dat. **b)** Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

VARIANTA 2 - *Rezolvare*

Subiectul 1

1.	Se accesează cifrele unităților și zecilor numărului memorat în variabila a folosind operatorii `div	/` și `mod	%.` Pentru precizarea răspunsului `d)` se acordă **4p.**
2.	Apelul recursiv `f(x div 10)| f(x/10)` pentru valorile x care au cifra unităților mai mică ca 5, sugerează că aceste cifre sunt ignorate. Cu restul cifrelor se contruiește un număr care are prima cifră `1` (prin apelul recursiv: `x%10+f(x/10)`, respectiv `x mod 10 +f(x div 10)*10`)). Pentru ca rezultatul să fie `1` trebuie ca toate cifrele lui x să fie strict mai mici decât `5`, iar cea mai mare valoare de `4` cifre cu toate cifrele pare este `4444`. Pentru răspunsul **c)** se acordă **4p.**
3.	Se generează în maniera "înaintare cu revenire", cuvintele: **abe**, **abec**, **abed**, **ace**, **aceb**, **aced**, etc. Pentru răspunsul **d)** se acordă **4p**.
4.	Din vectorul de referințe ascendente deducem că rădăcina arborelui este nodul **5** (**tata[5]=0**). Observăm că **tata[8]=9**, **tata[9]=7**, **tata[7]=2**, **tata[2]=5**, deci ascendenții nodului 8 sunt nodurile: **5,2,7,9.** Pentru precizarea răspunsului **d)** se acordă **4p**. 
5.	Graful, cu n=2021 noduri, fiind conex, nu poate avea noduri izolate, astfel gradul minim al unui nod poate fi 1. Gradul maxim poate fi n-1 (un nod poate fi adiacent cu toate celelalte n-1 noduri). Pentru precizarea răspunsului **d)** se acordă **4p**.

Subiectul 2

1.	**a)**	Se compară cifrele unităților numerelor **a** și **b**. Se alege cea mai mare dintre aceste cifre ale unităților, se adaugă la sfârșitul numărului **nr** și se elimină cifra aleasă din numărul din care provine. Se repetă acest procedeu până când unul dintre numerele **a** și **b** devine **0**. Pentru precizarea valorii **4351** se acordă **6p**.
	b)	Pentru variabila **a** poate fi citit orice număr natural de patru cifre care are ultima cifră egală cu **0**. Sunt **9** valori posibile pentru prima cifră, **10** valori posibile pentru a doua cifră și **10** valori posibile pentru a treia cifră. În total sunt **9x10x10=900** numere care pot fi citite pentru variabila **a**.

Pentru precizarea valorii **900** se acordă **4p**.

c) Înlocuirea structurii **cât timp a>0 și b>0 execută…** presupune utilizarea unei structuri repetitive **repetă...cât timp** sau **repetă...până când**:

```
┌dacă a>0 și b>0 atunci
| ┌repetă
| |┌dacă a %10 > b%10
| ||  atunci c←a%10
| ||         a←[a/10]
| ||  altfel c←b%10
| ||         b←[b/10]
| |└■
| |  nr←nr*10+c
| └cât timp a>0 și b>0
└■
```

```
┌dacă a>0 și b>0 atunci
| ┌repetă
| |┌dacă a %10 > b%10
| ||  atunci c←a%10
| ||         a←[a/10]
| ||  altfel c←b%10
| ||         b←[b/10]
| |└■
| |  nr←nr*10+c
| └până când a=0 sau b=0
└■
```

Fără testarea inițială a valorilor **a** și **b** (a>**0** și b>**0**), dacă una dintre ele este nulă, algoritmul obținut prin înlocuirea mecanică a structurii **cât_timp** cu structura **repetă…** va afișa răsturnatul celuilalt număr, spre deosebire de algoritmul din enunț care afișează valoarea 0, deci cei doi algoritmi nu sunt echivalenți.

Pentru alegerea unei structuri repetitive se acordă **2p.**, pentru testarea inițială a valoriilor lui **a** și **b** se acordă **2p.**, pentru testarea condiției de continuare sau oprire se acordă **1p**, iar pentru scrierea integrală a restului algoritmului se mai acordă **1p.** - în total **6p**.

d) Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și de decizie se acordă câte **1p**., pentru cele două instrucţiuni repetitive se acordă **3p**., iar pentru cele **6** atribuiri se acordă **2p**. Pentru structura corectă a programului se mai acordă **1p.**, în total **10p**.

2. Un răspuns corect: `abs(C.O.y)<=C.R`
Pentru orice răspuns corect se acordă **6p**.

3. Răspuns corect: `1 3 5 7 9 11`
La finalul executării secvenței date conținutul matricei A este:

```
6   5 4 3 2 1
7   6 5 4 3 2
8   7 6 5 4 3
9   8 7 6 5 4
10  9 8 7 6 5
11 10 9 8 7 6
```

Se acordă câte **1p**. pentru fiecare element indicat corect.

Subiectul 3

1. Prelucrarea oferită de subprogram determină poziția primei apariții a valorii minime din tablou, poziția ultimei apariții a valorile maxime în șir, apoi interschimbă cele două valori situate pe pozițiile determinate. Tabloul rezultat este returnat prin intermediul parametrului **v** al subprogamului.

	Limbajul Pascal
	see code below

```
Type vector=array[1..100]of integer;
procedure sub(n:byte;var v:vector);
var pmin,pmax,i:byte;
    aux:integer;
begin
        pmin:=1;
        pmax:=1;
        for i:=1 to n do
                if v[i]>=v[pmax] then
                        pmax:=i
                else if v[i]<v[pmin]
                        then pmin:=i;
        aux:=v[pmin];
        v[pmin]:=v[pmax];
        v[pmax]:=aux
end;
```

Limbajul C/C++

```
void sub(int n, int v[])
{
        int i, pmin=1, pmax=1,aux;
        for(i=2;i<=n;i++)
                if (v[i]>=v[pmax])
                        pmax=i;
                else
                        if (v[i]<v[pmin]) pmin=i;
        aux=v[pmin];
        v[pmin]=v[pmax];
        v[pmax]=aux;
}
```

Pentru antet corect se acordă **2p**. Pentru determinarea numărului minim din tabloul **v** se acordă **1p**, iar pentru determinarea poziției primei apariții a minimului se acordă **2p**. Pentru determinarea numărului maxim din tablou **v** se acordă **1p**, iar pentru determinarea poziției ultimei apariții a maximului se acordă **2p**. Pentru interchimbarea celor două numere determinate se acordă **1p**. Pentru corectitudinea globală a programului **1p**.

2. O soluție se poate obține separând cuvintele din text si verificând apariția primului cuvânt ca sufix pentru restul cuvintelor din text; în caz afirmativ, aceste cuvinte vor fi eliminate. Se poate contrui un șir nou care să conțină cuvintele ce vor rămâne în text, prin concatenarea acestora și inserarea caracterului * după fiecare cuvânt.

O altă soluție se poate obține, după separarea și eliminarea primului cuvânt din șir, prin determinarea tuturor aparițiilor primului cuvânt în șirul rămas. În situația în care caracterul * succede o astfel de apariție, atunci vom șterge din șir cuvântul corespunzător apariției.

Se pot utiliza funcţiilor predefinite pentru şiruri de caractere specifice limbajului.

Pentru declararea corectă a variabilelor se acordă **1p**. Pentru citirea șirului se acordă **1p**. Pentru separararea primului cuvânt din text se acordă **1p.** Pentru determinarea unui

cuvânt din text care admite ca sufix primul cuvânt și pentru eliminarea lui se acordă **2p**. Pentru eliminarea tuturor cuvintelor din șir cu proprietatea cerută se acordă **2p**. Pentru modificarea și afișarea șirului în formatul cerut se acordă **2p.** Pentru corectitudinea globală a programului se acordă **1p** - în total **10p**.

Limbajul Pascal	**Limbajul C/C++**

```pascal
var i,n,j,k :byte;
        s,p :string;
begin
  readln(s);
  n:=pos('*',s);
  p:=copy(s,1,n);
  delete(s,1,n);
  k:=pos(p,s);
  while k>0 do begin
    j:=k+n-1;
    i:=k;
    while(s[i]<>'*')and(i>0)
        do dec(i);
    inc(i);
    delete(s,i,j-i);
    k:=pos(p,s);
  end;
  s:='*'+s;
  writeln(s)
end.
```

```cpp
int main()
{
  char s[256], t[256], *p, *q, m;
  int i,j,n,ok;
  cin>>s;
  strcpy(t,"*");
  p=strtok(s,"*");
  n=strlen(p);
  q=strtok(NULL,"*");
  while(q)
  {
    m=strlen(q);
    ok=0;
    for(i=m-n;i>=0 && ok==0;i--)
      if(strcmp(p,q+i)==0)ok=1;
    if(ok==0)strcat(t,q);
    strcat(t,"*");
    q=strtok(NULL,"*");
  }
  strcpy(s,t); cout<<s;
  return 0;
}
```

3. **b)** Algoritmul de rezolvare propus determină suma cifrelor primului număr **sx** și apoi citește un număr din fișier, comparând suma cifrelor acestuia cu **sx**: în caz de egalitate se "prelungește" secvența curentă, în caz contrar, se trece la o nouă secvență cu o nouă sumă **sx** curentă. La finalul oricărei secvențe se analizează lungimea ei, în vederea memorării lungimii maxime.
Eficiența algoritmului, ca timp de executare, constă în faptul că determinăm valoarea cerută în timpul citirii datelor, deci parcurgem o singură dată șirul de numere. Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple, fără folosirea unor structuri suplimentare de date.
Pentru o descriere coerentă a metodei se acordă **1p.**, iar pentru justificarea eficienței, încă **1p**. - în total **2p**.

a)

Limbajul Pascal

```pascal
var f:text;
        nr,lgmax,lg,x,sx,snr:longint;
begin
        assign(f,'bac.in');reset(f);
        read(f,x);
```

```
          lg:=1;
          sx:=0;
          while x>0 do begin
                  sx:=sx+x mod 10;
                  x:=x div 10
          end;
          while not(eof(f)) do begin
                  read(f,nr); snr:=0;
                  while nr>0 do begin
                          snr:=snr+nr mod 10;
                          nr:=nr div 10
                  end;
                  if snr=sx then inc(lg)
                  else begin
                          if lgmax<lg then lgmax:=lg;
                          lg:=1; sx:=snr
                  end;
          end;
          if lgmax<lg then lgmax:=lg;
          close(f);
          write(lgmax)
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int main()
{       int lgmax=0, lg=1, nr, sx=0, x, snr;
        f>>x;
        while(x)
        {       sx=sx+x%10; x=x/10; }
        while(f>>nr)
        {       snr=0;
                while(nr)
                {   snr+=nr%10; nr=nr/10; }
                if (snr==sx) lg++;
                else {  if(lgmax<lg) lgmax=lg;
                        lg=1; sx=snr;
                }
        }
        if (lgmax<lg) lgmax=lg;
        cout<<lgmax;
        return 0;
}
```

Se acordă **1p**. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă **1p**. pentru citirea tuturor numerelor din fişier, **1p.** pentru un algoritm principial corect, încă **1p.** pentru determinarea valorii cerute, **1p.** pentru afișarea ei, **1p.** pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă **1p.** pentru alegerea unui algoritm eficient ca timp de executare (**O(n)**) și **1p.** pentru utilizarea eficentă a memoriei. În total, **8p**.

VARIANTA 3

SUBIECTUL 1 (20p)

Pentru itemul 1, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect.

1. Indicaţi expresia care are valoarea **`true`**/**1** dacă şi numai dacă numărul întreg memorat în variabila **x are exac trei cifre**. **(4p.)**

	Limbajul Pascal	Limbajul C/C++
a.	`x<1000`	`x<1000`
b.	`(100<=x) and (x<=999)`	`(100<=x) && (x<=999)`
c.	`(-999<=x) and (x<=999)`	`(-999<=x) && (x<=999)`
d.	`(-999<=x) and (x<=-100) or (100<=x) and (x<=999)`	`(-999<=x) && (x<=-100) || (100<=x) && (x<=999)`

2. Se consideră subprogramul **`f`** cu definiția următoare: **(4p.)**

Limbajul Pascal

```
procedure f(x:longint);
begin
        if x>9 then begin
                write(x mod
10);
                f(x div 10);
                write("*")
        end
        else write("#");
end;
```

Limbajul C/C++

```
void f(int x) {
        if(x>9) {
                printf("%d",x%10);/
cout<<x%10;
                f(x/10)
                printf("*"); /
cout<<"*";
        }
        else cout<<"#";
}
```

Indicați valoarea **`f(12345)`**

a. 5432#****	b. ****#5432	c. 5432####*	d. #****5432

3. În declararea următoare, câmpurile **`x`** şi **`y`** ale înregistrării pot memora coordonatele carteziene ale unui punct din planul **xOy**. Indicați expresia în limbajul **Pascal | C/C++** care să aibă valoarea **true|**1 dacă şi numai dacă punctul memorat în variabila **A** este situat pe axa **Ox**, iar punctul memorat în variabila **B** este situat pe axa **Oy**. **(4p.)**

Limbajul Pascal

```
type punct=record
 x,y:real
end;
var A,B:punct;
```

```
a. (A.y=0) and (B.x=0)
b. (A.y=0) or (B.x=0)
c. (A.x=0) or (B.y=0)
d. (A.x=0) and (B.y=0
```

Limbajul C/C++

```
struct punct
{ float x,y; } A,B;
```

```
a. (A.y==0) and (B.x==0)
b. (A.y==0) or (B.x==0)
c. (A.x==0) or (B.y==0)
d. (A.x==0) and (B.y==0
```

4. Care este numărul maxim de muchii pe care îl poate avea un graf neorientat cu **5** noduri? **(4p.)**			
a. `4`	**b.** `5`	**c.** `8`	**d.** `10`

5. Se consideră un arbore cu rădăcină reprezentat prin următorul vector "de taţi": `(0,1,1,1,4,4,6,7)`. Câte lanțuri elementare distincte, de lungime maximă, care au extremitatea finală în nodul `8`, există în arborele specificat. **(4p.)**			
a. 2	**b.** 1	**c.** 3	**d.** 0

SUBIECTUL 2 (40p)

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **`x%y`** restul împărţirii numărului întreg **`x`** la numărul întreg nenul **`y`** şi cu **`[a]`** partea întreagă a numărului real **`a`**.

```
citeşte x,y (numere naturale)
p←1
┌cât timp y>0 execută
| t←y
| ┌cât timp t>9 execută
| | t←[t/10]
| └■
| ┌dacă x%10 = t atunci p←p+1
| |  altfel scrie p
| |        p←1
| └■
| x←y
| citește y (număr natural)
└■
scrie p
```

a)	Scrieți ce se va afișa dacă se citesc, în această ordine, numerele: `12 23 34 592 13 39 91 1 0`? **(6p.)**
b)	Scrieţi un set de date de intrare pentru care algoritmul, în urma executării, va afișa valoarea `1234`. **(6p.)**
c)	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **`cât timp t>9 execută...`** cu o structură repetitivă de alt tip. **(6p.)**
d)	Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. Utilizând metoda backtracking se generează în ordine crescătoare toate numerele naturale de câte cinci cifre din mulţimea **`A={1,2,3,4}`**, numere în care suma primelor două cifre este **`5`** .Primele numere generate sunt, în ordine: **`14111`**, **`14112`**, **`14113`**, **`14114`**, **`14121`**, **`14122`**,... Scrieți câte dintre numerele generate încep și se termină cu cifra `3`? **(6p.)**

3. Variabila S memorează un șir cu maximum 20 de caractere. Scrieți ce se afișează în urma executării secvenței: **(6p.)**

Limbajul Pascal	Limbajul C/C++
S:= ‘3a+b=9’; **S:=copy(S,2,5);** **delete (S,2,2);** **write (S);**	**char S[20]=”3a+b=9”;** **strcpy(S,S+1);** **strcpy(S+1,S+3);** **cout<<S;**

SUBIECTUL 3 (30p)

1. Scrieţi definiţia completă a unui subprogram **sub**, cu trei parametri, care primeşte prin intermediul parametrilor:

- **x**, un număr natural, având cel mult patru cifre;
- **n**, un număr natural nenul (0 < **n** < 100);
- **v**, un tablou unidimensional care memorează un şir de **n** numere naturale, fiecare având cel mult patru cifre;

Subprogramul **sub** determină dublarea fiecărei apariții a valorii parametrului x în șirul de numere primit prin parametrul **v**. De asemenea subprogramul va furniza prin intermediul parametrului **n** numărul de numere din şirul modificat, iar prin parametrul **v** tabloul modificat. Dacă valoarea parametrului x nu apare în șirul de numere atunci valorile parametrilor nu se vor modifica.

Exemplu. Pentru valorile **n=10**, **v=(51, 19, 3, 52, 19, 215, 19, 19, 4, 65)**, **x=19** ale parametrilor, în urma apelului, subprogramului **sub** va furniza prin parametrii indicaţi valorile **n=14** şi **v=(51, 19, 19, 3, 52, 19, 19, 215, 19, 19, 19, 19, 4, 65).**

(10p.)

2. Să se construiască o matrice cu **n** linii şi **m** coloane care conţine:

- pe prima coloană, de sus în jos, toate numerele naturale de la **1** la **n**, în ordine crescătoare;

- pe prima linie, de la stânga la dreapta toate numerele naturale de la 1 la **m**, în ordine crescătoare;

Restul elementelor se vor inițializa cu suma elementelor vecine situate în stânga și deasupra elementului curent, ca în exemplu.

Scrieţi programul care citeşte de la tastatură două valori naturale **n** și m (**2<n<25, 2<m<25**), construieşte matricea conform cerinţei şi o afişează pe ecran, pe linii, cu spaţii între elementele de pe fiecare linie.

De exemplu, dacă se citeşte de la tastatură **n=3** și **m=5**, se va construi şi se va afişa pe ecran matricea:

```
1 2 3 4 5
2 4 7 11 16
3 7 14 25 41
```

(10p.)

3. Fişierul text **BAC.TXT** conţine un şir **s** de cel mult un milion de numere naturale, formate fiecare din cel mult **9** cifre, separate prin câte un spaţiu

a) Scrieţi un program **C/C++** care citeşte numerele din fişier şi determină, utilizând un algoritm eficient din punct de vedere al memoriei utilizate şi al timpului de executare, lungimea secvenţei obţinute prin eliminarea din cele două extremităţi ale şirului **s** a unui număr minim de numere, fără

a schimba ordinea celorlalte numere, astfel încât secvenţa rezultată să înceapă cu un număr format doar din cifre pare şi să se termine cu un număr format doar din cifre impare. Programul va afişa pe ecran lungimea secvenţei obţinute.

De exemplu, dacă fişierul **BAC.TXT** conţine numerele:

`132 214 62 34 28 34 8 45 18 72 35 12 17 34 4 135 63 81 101`

pe ecran se va afişa numărul `14`, deoarece secvenţa căutată se obţine prin eliminarea numerelor subliniate şi este formată din 14 numere **(8p.)**

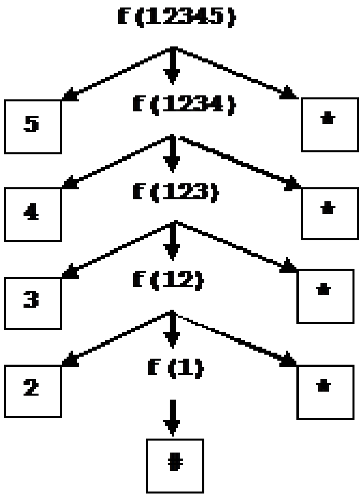
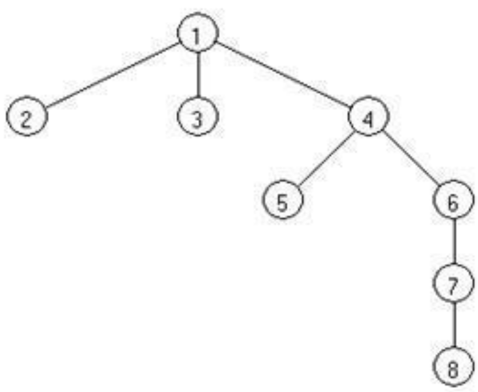
b) Descrieţi în limbaj natural algoritmul proiectat justificând eficiența acestuia. **(2p.)**

⚠ **Observaţii:**

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 3 - *Rezolvare*

Rezolvare - Subiectul 1

1.	Valoarea variabilei **x** trebuie să satisfacă condiția `100<=|x|<=999.` Pentru precizarea răspunsului d**)** se acordă **4p**.
2.	În desenul alăturat este prezentat mecanismul prin care se execută apelul `f(12345)` al subprogramului recursiv **f**. Se vor afişa, în ordine: **`5432#****,`** răspuns pentru care se acordă **4p.** 
3.	Răspuns corect **a.** pentru care se acordă **4p.**
4.	Pentru răspunsul **d)** se acordă **4p.**
5.	Cele mai lungi lanțuri elementare din arbore cu o extremitate în nodul 8 sunt: `[2,1,4,6,7,8]` și `[3,1,4,6,7,8]`.  Răspuns corect **a.** pentru care se acordă **4p.**

Rezolvare - Subiectul 2

1.	**a)**	Șirul de date de intrare (fără valoarea 0 utilizată pentru încheierea operației de citire) se împarte în secvențe formate din numere cu proprietatea că prima cifră a numărului curent este egală cu cifra unităților numărului precedent, iar cifra unităților numărului curent este egală cu prima cifră a numărului următor. Astfel șirul de date de intrare se împarte în trei secvențe de acest tip cu lungimile 3,1,4: **`12 23 34 592 13 39 91 1 0`** Pentru precizarea valorii **`314`** se acordă **6p**.
	b)	Un răspuns posibil poate fi setul de date de intrare: 1 2 2 3 3 3 4 4 4 4 0 Pentru orice set de date de intrare corect se acordă **6p**.
	c)	Înlocuirea structurii **`cât timp t>0 execută ...`** presupune utilizarea unei structuri repetitive **`repetă - cât timp`** sau **`repetă - până când`** : `┌repetă` `│ t←[t/10]` `└cât timp t>9` `┌repetă` `│ t←[t/10]` `└până când t<=9` Pentru alegerea unei structuri repetitive se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 2p, iar pentru scrierea integrală a restului algoritmului se mai acordă 2p. - în total **6p**.
	d)	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc. Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului se acordă câte 1p, pentru instrucțiunea de decizie decizie se acordă 1p., pentru instrucţiunea repetitivă se acordă 3p., iar pentru cele 6 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.
2.		Numerele cerute sunt de forma: 32xx3 cu x϶{1,2,3,4}. Sunt generate 4x4=16 numere cu proprietatea din enunţ. Pentru răspunsul corect se acordă **6p.**
3.		Șirului de caractere S i se atribuie valoarea 3a+b. Din acesta se șterge caracterul de pe prima poziție și noul șir va fi a+b=9. Instrucțiunea următoare șterge caracterul al doilea și al treilea. Rezultatul afișat este a=9. Pentru răspunsul corect se acordă **6p.**

Rezolvare - Subiectul 3

1. Prelucrarea oferită de subprogram determină fiecare apariție a valorii **x** în tablou. Dublarea unui element cu valoarea **x** se realizează prin deplasarea cu o poziție la dreapta a tuturor elementelor situate la dreapta celui ce urmează a fi dublat, începând de la sfârșitul șirului. După realizarea deplasării, se mărește valoarea lui **n** cu 1.

Limbajul Pascal	Limbajul C/C++
`Type vector=array[1..100]of integer;` `procedure sub(x:integer;var n:byte;var v:vector);` `var i,j:byte;` `begin` `  i:=1;` `  while i<=n do begin` `   if v[i]=x then begin` `    for j:=n downto 1 do` `     v[i+1]:=v[i];` `    n:=n+1;` `    i:=i+1` `   end;` `   i:=i+1` `  end;` `end;`	`void sub(int x,int &n,int v[])` `{` `  int i=1,j;` `  while(i<=n)` `  {` `   if(v[i]==x) {` `     for(j=n;j>=1;j--)` `       v[j+1]=v[j];` `     n++;` `     i++;` `   }` `   i++;` `  }` `}`

Pentru antet corect se acordă 3p. (Pentru respectarea structurii antetului (**procedure/void**) se acordă 1p., pentru declararea corectă a parametrilor de intrare **x**, pentru declararea corectă a parametrilor **n** și **v** încă 1p., pentru declararea variabilelor locale încă 1p).
Pentru determinarea fiecărei apariţii ale numărului **x** în tabloul 1p. Pentru dublarea primei apariții în tabloului și incrementarea valorii parametrului **n** se acordă câte 1p. Pentru determinarea tuturor aparițiilor lui **x** în tablou se acordă 2p. Pentru mărirea corespunzătoare a valorii parametrului **n** se acordă 1p. Pentru corectitudinea globală a programului 1p.
În total **10p.**

2. Pot fi concepute mai multe metode de obținere a matricei cerute. Metoda clasică ce completează coloanele matricei, cu liniile, respectiv coloanele, numerotate de la **1** la **n**, respectiv de la **1** la **m**, în maniera descrisă, este:

```
┌pentru i←1,n execută
│┌pentru j←1,m execută
││ ┌dacă i=1 atunci A[i,j]←j
││ │ altfel ┌dacă j=1 atunci A[i,j]←i
││ │        │  altfel A[i,j]←A[i,j-1]+A[i-1][j]
││ │        └■
││ └■
│└■
└■
```

Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui n și m se mai acordă 1p., pentru completarea corectă a primei linii se acordă 1p., pentru completarea corectă a primei coloane se acordă 1p., pentru completarea corectă a celorlalte elemente se acordă 3p., iar pentru afișarea matricei conform cerinței se acordă 2p. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă 1p. - în total **10p**.

3.

b. Algoritmul de rezolvare propus presupune determinarea poziției pp a primului număr format doar din cifre pare, respectiv ui a ultimului număr format doar din cifre impare, din fişier pe măsura citirii datelor și utilizează doar variabile simple fără folosirea unor structuri suplimentare de date.

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul cu un număr impresionant de numere. Deoarece numărul de valori din fișier (1000000) este dimensiunea care contează în problemă, spunem că am obținut un algoritm liniar. Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple.

Lungimea subşirului căutat va fi: **ui-pp+1** sau **0** dacă **pp=0 sau ui=0**.

Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. - în total **2p.**

a.

Limbajul Pascal

```
var f:text;
    nr, i,pp, ui,p, imp:longint;
begin
 assign(f,'bac.in'); reset(f);
 while not(eof(f)) do begin
        read(f,nr);
        inc(i);
        p:=0;
        imp:=0;
        repeat
         if nr mod 2=0 then inc(p)
         else inc(imp);
         nr:=nr div 10
        until nr=0;
        if (imp=0) and (pp=0)
         then pp=i
          else if (p==0) ui:=i;
 end;
 if pp*up=0 then writeln ('0')
 else writeln(ui-pp+1);
 close(f);
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int main()
{
  int pp=0, ui=0, i=0, nr, p, imp;
  while(f>>nr)
  {     p=imp=0;
        i++;
        do
        { if(nr%2==0) p++;
          else imp++;
          nr/=10;
        }while(nr>0);
        if(imp==0 && pp=0)pp=i;
         else if (p==0) ui=i;
   }
   if(pp*ui) cout<<"Nu exista";
   else cout<<ui-pp+1;
   f.close();
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea valorii cerute, 1p. pentru afișarea ei, 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare ($O(n)$) și 1p. pentru utilizarea eficientă a memoriei. În total **8p.**

VARIANTA 4

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte		
1.	Considerând o variabilă **x** de tip real, indicați expresia care are valoarea **true/1** dacă şi numai dacă **x∈[1,5]U[7,9]**.	
	Limbajul Pascal	**Limbajul C/C++**
a.	`((x>=1) or (x<=9)) and ((x<=5) or (x>=7))`	`(x>=1 || x<=9) && (x<=5 || x>=7)`
b.	`not((x>0) and (x<7)) and (x>=1)`	`!(x>5 && x<7) && x>=1`
c.	`(x>=1) and (x<=5) and (x>=7) and (x<=9)`	`x>=1 && x<=5 && x>=7 && x<=9`
d.	`not((x<1) or ((x>5) and (x<7)) or (x>9))`	`!(x<1 || (x>5 && x<7) || x>9)`
2.	Se consideră tipul de date **Data**, capabil să memoreze o dată calendaristică şi tipul de date **Elev**, capabil să memoreze numele, data nașterii și media anuală a unui elev. Care dintre următoarele expresii are valoare **true/1** dacă şi numai dacă elevul ale cărui date sunt memorate în variabila **e** de tipul **Elev** este născut în anul **2000**?	
	Limbajul Pascal	**Limbajul C/C++**
	`type Data = record` `zi, luna, an:integer;` `end;` `type Elev = record` `nume:string[30];` `data_nasterii:Data;` `media:real;` `end;` `var e:Elev;`	`struct Data` `{` `int zi, luna, an;` `};` `struct Elev` `{` `char nume[30];` `Data data_nasterii;` `Float media;` `};` `Elev e;`
a.	`e^.data_nasterii^.an = 2000`	`e->data_nasterii->an == 2000`
b.	`e.data_nasterii.an = 2000`	`e.data_nasterii.an == 2000`
c.	`e.an = 2000`	`e.an == 2000`
d.	`data_nasterii.an = 2000`	`data_nasterii.an == 2000`
3.	Utilizând metoda backtracking se generează, în ordinea crescătoare a valorii, toate numerele formate din exact **n** cifre nenule distincte şi având suma cifrelor egală cu **s**. Astfel, pentru **n=2** și **s=10**, se obțin, în ordine, numerele: **19,28,37,46,64,73,82,91**. Folosind aceeași metodă, se generează numerele formate din **n=4** cifre distincte și având suma cifrelor **s=25**. Care va fi al treisprezecelea număr generat?	
	a. 3598 b. 2986 c. 3589 d. 3679	
4.	Fie **G** un graf neorientat complet cu **10** vârfuri. Numărul maxim de muchii care pot fi eliminate din **G** astfel încât acesta să rămână conex este:	
	a. 9 **b.** 32 **c.** 36 **d.** 40	

5.	Se consideră un graf neorientat **G** cu **6** noduri etichetate cu numerele naturale de la **1** la **6**, dat prin listele de adiacență de mai jos: **1: 2,6** **2: 1,3** **3: 2,4,5,6** **4: 3,5** **5: 3,4** **6: 1,3** Care dintre următoarele afirmații referitoare la graful dat G este adevărată?
a.	gradul oricărui vârf este un număr par nenul
b.	este aciclic
c.	nu este conex
d.	conține vârfuri cu gradul 0

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**
S-a notat cu **x%y** restul împărțirii numărului întreg **x** la numărul întreg nenul **y**, iar cu **[x]** partea întreagă a numărului real **x**.

```
citeşte a,b (numere naturale nenule, a<=b)
nr ← 0
┌pentru i ← a,b execută
│ d ← 0
│┌pentru j ← 2,[i/2] execută
││┌dacă i%j = 0 atunci
│││  d ← j
││└■
│└■
│┌dacă d = 0 atunci
││  nr ← nr+1
│└■
└■
scrie nr
```

a.	Ce valoare va fi afişată dacă se citesc valorile **7** şi **13**? **(6p.)**
b.	Pentru **a = 10**, scrieţi cea mai mare valoare care poate fi introdusă pentru **b** astfel încât rezultatul afişat să fie **4**. **(6p.)**
c.	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **pentru j ← 2,[i/2] execută** cu o structură repetitivă de alt tip. **(10p.)**
d.	Scrieţi programul **Pascal/C/C++** corespunzător algoritmului dat. **(6p.)**

2. Se consideră două şiruri de caractere **s** şi **t** declarate prin:

Limbajul Pascal	Limbajul C/C++
`var s,t:string[30];`	`char s[31],t[31];`

Ştiind că ambele şiruri sunt formate numai din litere mici ale alfabetului englez, scrieţi **doar** instrucţiunea sau instrucţiunile care afişează pe ecran mesajul **DA** în cazul în care șirul **s** este un prefix al șirului **t** sau mesajul **NU** în caz contrar. **(6p.)**

3.	Funcţia **f** este definită astfel:

Limbajul Pascal	Limbajul C/C++
<pre>function f(n:longint; p:longint):longint; begin if(p > n div 2) then f := 0 else if(n mod p = 0) then f := p + f(n div p, p+1) else f := f(n, p+1); end;</pre>	<pre>long f(long n, long p) { if(p > n/2) return 0; else if(n%p == 0) return p + f(n/p, p+1); else return f(n, p+1); }</pre>

Scrieţi valoarea care se obţine în urma apelului **f(24,2)**. **(6p.)**

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Scrieți definiția completă a subprogramului **cifre**, cu doi parametri, care primeşte prin intermediul parametrului **a** un număr natural format din maxim **9** cifre nenule și furnizează prin al doilea parametru **b** numărul obţinut prin eliminarea cifrelor lui **a** aflate pe poziții pare. Cifrele numărului **a** se numerotează de la dreapta spre stânga, începând cu poziția **0** (corespunzătoare cifrei unităților).

Exemplu: pentru **a=2334157**, valoarea returnată prin **b** va fi **345.** **(10p.)**

2. Să se construiască o matrice cu **n** linii şi **n** coloane care conţine pe prima linie numerele naturale de la **1** la **n** în ordine crescătoare, pe a doua linie numerele naturale de la **n+1** la **2n** în ordine descrescătoare, pe a treia linie numerele naturale de la **2n+1** la **3n** în ordine crescătoare, pe a patra linie numerele naturale de la **3n+1** la **4n** în ordine descrescătoare şi aşa mai departe până la linia **n**. Scrieţi programul care citeşte de la tastatură o valoare naturală **n** (**2<=n<=50**), construieşte matricea conform cerinţei şi o afişează pe ecran, pe linii, cu spaţii între elementele de pe fiecare linie.
De exemplu, dacă se citeşte de la tastatură **n=5**, se va construi şi se va afişa matricea:

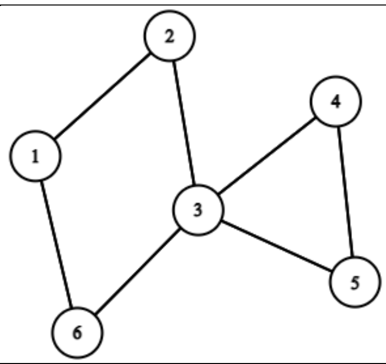
```
1 2 3 4 5
10 9 8 7 6
11 12 13 14 15
20 19 18 17 16
21 22 23 24 25
```

(10p.)

3. Fișierul text **bac.in** conține, pe prima linie, cel mult **1000000** de numere naturale nenule, fiecare fiind format din cel mult **9** cifre. Oricare două numere aflate pe poziții consecutive sunt despărțite printr-un spațiu.
a) Scrieți un program care, folosind un algoritm eficient din punct de vedere al timpului de executare și al spațiului de memorie utilizat, determină și scrie în fișierul **bac.out** cel mai mare număr natural care se poate obține din cifrele tuturor numerelor din fișierul **bac.in**. **(8p.)**
Exemplu: dacă fișierul **bac.in** conține numerele **2117 90 885 515 37,** atunci fișierul **bac.out** trebuie să conțină numărul **98877555321110.**
b) Descrieți în limbaj natural metoda utilizată și explicați în ce constă eficiența ei. **(2p.)**

VARIANTA 4 - *Rezolvare*

Subiectul 1

1.	Un număr real $x \in [1,5] \cup [7,9]$dacă și numai dacă $x \notin (-\infty, 1) \cup (5,7) \cup (9, +\infty)$, deci răspunsul corect este **d)**.
2.	Deoarece variabila **`e`** de tip **`Elev`** conține data nașterii elevului respectiv în câmpul **`data_nașterii`** de tip **`Data`**, rezultă că data nașterii elevului respectiv (adică ziua, luna și anul) poate fi accesată prin expresia **`e.data_nașterii`**, deci anul nașterii sale poate fi accesat prin expresia **`e.data_nașterii.an`**. În concluzie, răspunsul corect este **b)**.
3.	Folosind metoda backtracking, primele **`13`** numere formate din **`n=4`** cifre distincte și având suma cifrelor **`s=25`** care se vor genera sunt `1789, 1798, 1879, 1897, 1978, 1987, 2689, 2698, 2869, 2896, 2968, 2986` și **`3589`**, deci răspunsul corect este **c)**.
4.	Deoarece graful neorientat G este complet și are $n = 10$ vârfuri, înseamnă că el este conex și are $\frac{n(n-1)}{2} = 45$ muchii. Deoarece un graf conex minimal (adică un graf care prin eliminarea unei singure muchii nu mai este conex) este un arbore, rezultă că graful G trebuie să conțină cel puțin $n - 1$ muchii (orice arbore cu n vârfuri are $n - 1$ muchii!), adică se pot elimina maxim $45 - 9 = 36$ de muchii, deci răspunsul corect este **c)**.
5.	În figura alăturată este reprezentat graful dat în enunț. Se observă ușor faptul că răspunsul corect este **a)**. 

Subiectul 2

1.	**a)**	Algoritmul calculează câte numere prime există între **`a`** și **`b`**, deoarece variabila **`d`** va conține fie cel mai mare divizor propriu al unui număr **`i`** cuprins între **`a`** și **`b`**, fie valoarea `0` dacă numărul **`i`** nu are niciun divizor propriu (ceea ce înseamnă că numărul **`i`** este prim!). Pentru precizarea valorii `3` (între `7` și `13` există `3` numere prime: `7`, `11` și `13`) se acordă **6p**., iar pentru orice altă valoare se acordă **0p**.
	b)	Deoarece **`a=10`**, înseamnă că trebuie determinată cea mai mare valoare posibilă pentru **`b`** astfel încât între `10` și **`b`** să existe **4** numere prime. Se observă ușor faptul că între `10` și `19` există **4** numere prime (`11`, `13`, `17` și `19`), numerele `20`, `21` și `22` nu sunt prime, iar numărul `23` este prim, deci cea mai mare valoare posibilă pentru **`b`** astfel încât între `10` și **`b`** să existe **4** numere prime este `22`. Pentru precizarea valorii `22` se acordă **6p**., iar pentru orice altă valoare se acordă **0p**.

	c)	Structura repetitivă pentru **`j<-2,[i/2] execută`** se poate înlocui cu una dintre structurile repetitive **`cât timp... execută`** sau **`repetă... până când`**, având grijă la inițializarea, testarea și incrementarea variabilei **`j`**:

```
citeşte a,b
nr ← 0
┌pentru i ← a,b execută
| d ← 0
| j ← 2
|┌cât timp j<=[i/2] execută
|| ┌dacă i%j = 0 atunci
|| |  d ← j
|| └■
|| j ← j+1
|└■
|┌dacă d = 0 atunci
||  nr ← nr+1
|└■
└■
scrie nr
```

```
citeşte a,b
nr ← 0
┌pentru i ← a,b execută
| d ← 0
| j ← 2
|┌repetă
|| ┌dacă i%j = 0 atunci
|| |  d ← j
|| └■
|| j ← j+1
|└până când j > [i/2]
|┌dacă d = 0 atunci
||  nr ← nr+1
|└■
└■
scrie nr
```

Atragem atenția asupra faptului că structura repetitivă **`cât timp...execută`** asigură echivalența cu structura **`pentru...execută`** în toate situațiile, pe când structura **`repetă...până când`** este echivalentă doar în unele situații, printre care și cea din acest algoritm!

Pentru alegerea unei structuri repetitive se acordă **2p.**, pentru inițializarea lui **`j`** se acordă **1p.**, pentru testarea condiției de continuare sau oprire se acordă **3p.**, pentru incrementarea lui **`j`** se acordă **1p.**, iar pentru scrierea integrală a restului algoritmului se mai acordă **3p.**, deci se obțin, în total, **10p.**

d) Se testează capacitatea de a reprezenta pe hârtie algoritmul dat folosind un limbaj de programare studiat. Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afișare a rezultatului și de decizie se acordă câte **0.5p.**, pentru cele două instrucțiuni repetitive se acordă **2p.**, iar pentru cele 4 atribuiri se acordă **1p.** Pentru structura corectă a programului se mai acordă **1p.**, deci se obțin, în total, **6p.**

2. Pentru a testa dacă șirul **`s`** este un prefix al șirului **`t`** trebuie să verificăm dacă șirul **`s`** apare ca subșir în șirul **`t`** începând cu prima poziție (în limbajul Pascal) sau de la adresa primului caracter (în limbajele C/C++):

Limbajul Pascal	Limbajul C++
`if pos(s,t) = 1 then` `    writeln('DA')` `else` `    writeln('NU');`	`if(strstr(t, s) == s)` `  cout<<"DA";` `else` `  cout<<"NU";`

Pentru utilizarea funcției corespunzătoare se acordă **3p.**, pentru afișarea șirului de caractere cerut se acordă **2p.**, iar pentru corectitudinea sintactică a secvenței se mai acordă **1p.**, deci se obțin, în total, **6p**.

3.	În urma apelului **`f(24,2)`** se va obține valoarea **5**, deoarece **`f(24,2) = 2 + f(12,3) = 2 + 3 + f(4,4) = 2 + 3 + 0 = 5`**. Pentru precizarea valorii **5** se acordă **6p**., iar pentru orice altă valoare se acordă **0p**.

Subiectul 3

1. Subprogramul construiește valoarea cerută cifră cu cifră, adăugând la numărul **b** (inițial egal cu **0**) doar cifrele numărului **a** care se găsesc pe poziții impare. Numerotarea cifrelor lui **a** se realizează de la dreapta spre stânga, folosind variabila auxiliară **p**, începând cu poziția **0** (corespunzătoare cifrei unităților). Deoarece cifrele lui **a** sunt accesibile de la ultima către prima, trebuie utilizate înmulțiri cu puteri consecutive ale lui **10**.

Limbajul Pascal

```
procedure cifre(a:longint;
                var b:longint);
var uc,p:byte; z:longint;
begin
      b:=0;
      z:=1;
      p:=0;
      while a>0 do
      begin
            uc:=a mod 10;
            if p mod 2=1 then
            begin
                  b:=b+uc*z;
                  z:=z*10
            end;
            a:=a div 10;
            p:=p+1
      end
end;
```

Limbajul C++

```
void cifre(int a,int &b)
{
      int uc,p,z;
      b=0;
      z=1;
      p=0;
      while(a>0)
      {
            uc=a%10;
            if(p%2==1)
            {
                  b=b+uc*z;
                  z=z*10;
            }
            a/=10;
            p++;
      }
}
```

Pentru respectarea structurii antetului (**procedure**/**void**) se acordă **1p.**, pentru declararea corectă a parametrului de intrare **a** încă **1p.**, pentru declararea corectă a parametrului de ieșire **b** încă **1p.**, pentru declararea variabilelor locale încă **1p.**, iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă **1p.** Corectitudinea algoritmică a prelucrării în vederea obținerii valorii cerute este notată cu **4p.** Pentru transmiterea naturală a rezultatului prin parametrul **b** se acordă **1p**, aceasta însemnând că valoarea lui **b** nu se afișează și nu se returnează explicit. Astfel, în total, se obțin **10p.**

2. Există mai multe metode de obținere a matricei cerute, una dintre ele fiind următoarea (am considerat faptul că liniile și coloane matricei sunt numerotate de la **0**):

```
k ← 1
┌pentru i ← 0,n-1 execută
│┌dacă i%2 = 0
││  atunci
││    ┌pentru j ← 0,n-1 execută
││    │  a[i][j] ← k
││    │  k ← k+1
││    └■
││  altfel
││    ┌pentru j ← n-1,0,-1 execută
││    │  a[i][j] ← k
││    │  k ← k+1
││    └■
│└■
└■
```

Pentru declararea corectă a matricei se acordă **1p.**, pentru citirea lui **n** se mai acordă **1p.**, pentru parcurgerea unei linii se acordă **1p.**, pentru completarea corectă a unei linii (pară+impară) se acordă **1p.**+**1p.**, pentru completarea corectă a tuturor liniilor (pare+impare) se acordă **1p.**+**1p.**, iar pentru afișarea matricei conform cerinței se acordă **2p.** În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă **1p.**, deci se obțin, în total, **10p**.

3. **b)** Vom utiliza un vector de frecvențe **fc** cu **10** elemente pentru a contoriza aparițiile fiecărei cifre. Vom parcurge fișierul **bac.in** caracter cu caracter, folosind variabila **c** de tip **char**, și, în cazul în care variabila **c** conține o cifră, vom crește frecvența sa cu **1**. Cel mai mare număr care se poate forma folosind cifrele tuturor numerelor din fișier se obține scriind în fișierul **bac.out** fiecare cifră **c**, de la **9** la **0**, de un număr de ori egal cu frecvența sa.

Eficiența algoritmului din punct de vedere al timpului de executare constă în faptul că parcurgem o singură dată fișierul de intrare **bac.in**, obținând astfel un algoritm liniar în raport cu numărul de cifre din fișier. Eficiența algoritmului din punct de vedere al memoriei utilizate constă în faptul că se va folosi doar un vector de frecvențe cu **10** componente, evitând astfel memorarea tuturor celor maximum **9000000** de cifre din fișier într-un vector de caractere.

Pentru o descriere coerentă a metodei se acordă **1p.**, iar pentru justificarea eficienței, încă **1p.**, deci se obțin, în total **2p.**

a) **Limbajul Pascal**

```
var fin,fout:text;
      c:char;
      i,j:longint;
      fc:array[0..9]of byte;
begin
      for i:=0 to 9 do
        fc[i]:=0;

      assign(fin,'bac.in');
      reset(fin);

      while(not eof(fin)) do
      begin
```

```
        read(fin,c);
        if((c>='0')and(c<='9')) then
            inc(fc[ord(c)-ord('0')]);
      end;

      close(fin);

      assign(fout,'bac.out');
      rewrite(fout);

      for i:=9 downto 0 do
        for j:=1 to fc[i] do
            write(fout,i);

      close(fout);
end.
```

Limbajul C++

```
#include <fstream>

using namespace std;

ifstream fin("bac.in");
ofstream fout("bac.out");

int main()
{
      int fc[10],i,j;
      char c;

      for(i=0;i<10;i++)
        fc[i]=0;

      while(fin>>c)
        if((c>='0')&&(c<='9'))
            fc[c-'0']++;

      fin.close();

      for(i=9;i>=0;i--)
        for(j=0;j<fc[i];j++)
            fout<<i;

      fout.close();

      return 0;
}
```

Se acordă **1p.** pentru operațiile cu fișiere (declarare, nume corect și deschidere pentru citire), încă **1p.** pentru citirea tuturor cifrelor din fișier, **1p.** pentru un algoritm corect, încă **2p.** pentru determinarea numărului cerut și **1p.** pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).
Se acordă **1p.** pentru alegerea unui algoritm eficient ca timp de executare și **1p.** pentru utilizarea eficientă a memoriei, deci se obțin, în total, **8p.**

VARIANTA 5

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Fie **n** o variabilă de tip întreg care conține un număr natural format din exact **3** cifre. Indicați expresia care are valoarea **true/1** dacă și numai dacă prima cifră a numărului **n** este egală cu ultima.

	Limbajul Pascal	Limbajul C/C++
a.	`n div 10 = n mod 100`	`n/10 == n%100`
b.	`n mod 10 = n div 100`	`n%10 == n/100`
c.	`n div 10 = n div 100`	`n/10 == n/100`
d.	`n mod 10 = n mod 100`	`n%10 == n%100`

2. Se consideră tipul de date **Punct2D**, capabil să memoreze coordonatele unui punct în plan, şi tipul de date **Punct3D**, capabil să memoreze coordonatele unui punct în spațiu. Care dintre următoarele secvențe de instrucțiuni poate fi folosită pentru a inițializa toate coordonatele punctului **q** cu valoarea **0**?

Limbajul Pascal

```
type Punct2D=record
   x,y:real;
end;

type Elev=record
   p:Punct2D;
   z:real;
end;

var q:Punct3D;
```

Limbajul C/C++

```
struct Punct2D
{
   double x,y;
};

struct Punct3D
{
   Punct2D p;
   double z;
};

Punct3D q;
```

	Limbajul Pascal	Limbajul C/C++
a.	`q.x:=0; q.y:=0; q.z:=0;`	`q.x = q.y = q.z = 0;`
b.	`q.p:=0; q.z:=0;`	`q.p = q.z = 0;`
c.	`q.p:=0; q.z:=0;`	`q.p = q.z = 0;`
d.	`q.p:=0; q.z:=0;`	`q.p = q.z = 0;`

3. Utilizând metoda backtracking se generează, în ordinea crescătoare a valorii, toate numerele formate din exact **n** cifre aflate în ordine strict crescătoare şi având prima cifră cel puțin egală cu **c**. Astfel, pentru **n=3** şi **c=5**, se obţin, în ordine, numerele: **567**, **568**, **569**, **578**, **579**, **589**, **678**, **679**, **689** şi **789.** Folosind aceeaşi metodă, se generează numerele formate din **n=5** cifre aflate în ordine strict crescătoare şi având prima cifră cel puțin egală cu **c=3**. Care va fi al zecelea număr generat?

4. Un graf neorientat **G** are **6** vârfuri, numerotate de la **1** la **6**, și muchiile **(1,2)**, **(2,3)**, **(2,6)**, **(6,1)**, **(6,5)**, **(3,6)**, **(4,5)**, **(3,4)**, **(6,4)** şi **(3,1)**. Numărul minim de muchii care pot fi eliminate din **G** astfel încât gradele tuturor vârfurilor să fie egale este:

a. 5	**b.** 6	**c.** 3	**d.** 4

5.	Se consideră un arbore cu rădăcină având **8** noduri, etichetate cu numerele de la **1** la **8**, dat prin următorul vector de taţi: **(5,3,8,8,0,1,5,5)**. Care dintre următoarele noduri nu este o frunză?		
a. 3	**b.** 4	**c.** 6	**d.** 7

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**
S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y**, iar cu **[x]** partea întreagă a numărului real **x**.

```
citeşte a,b (numere naturale nenule, a<=b)
nr ← 0
┌pentru i ← a,b execută
│ d ← 1
│ aux ← i
│┌cât timp aux≠0 execută
││  c ← aux%10
││ ┌dacă c%2 = 1 atunci
││ │  d ← 0
││ └■
││ aux ← [aux/10]
│└■
│┌dacă d = 1 atunci
││  nr ← nr+1
│└■
└■
scrie nr
```

a. Ce valoare va fi afişată dacă se citesc valorile **201** şi **208**? **(6p.)**

b. Pentru **a=818**, scrieţi cea mai mare valoare care poate fi introdusă pentru **b** astfel încât rezultatul afişat să fie **7**. **(6p.)**

c. Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **pentru...execută** cu o structură repetitivă de alt tip. **(10p.)**

d. Scrieţi programul **Pascal/C/C++** corespunzător algoritmului dat. **(6p.)**

2. Considerăm următoarea secvență de program:

Limbajul Pascal	Limbajul C++
`var s:string[20];` `...................` `s:='examene';` `s:=s+copy(s,3,3);` `s:=copy(s,5,20);` `writeln(length(s));`	`char s[20];` `...................` `strcpy(s,"examene");` `strncat(s,s+2,3);` `strcpy(s,s+4);` `cout<<strlen(s);`

Scrieţi ce se va afișa pe ecran în urma executării secvenței de instrucțiuni de mai sus. **(6p.)**

3.	Funcţia **f** este definită astfel:	
	Limbajul Pascal	**Limbajul C/C++**
	<pre>function f(n:longint; x:byte):byte; begin if(n<10) then if(n=x) then f:=0 else f:=1 else if(n mod 10 = x) then f:=f(n div 10, x) else f:=1+f(n div 10, x); end;</pre>	<pre>long f(long n, int x) { if(n<10) if(n==x) return 0; else return 1; else if(n%10==x) return f(n/10, x); else return 1+f(n/10, x); }</pre>
	Scrieţi valoarea care se obţine în urma apelului **f(100100,1)**. **(6p.)**	

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Scrieţi definiţia completă a subprogramului **divp**, cu **3** parametri, care primeşte prin intermediul parametrului **a** un număr natural cuprins între **2** și **1000000000** și furnizează prin al doilea parametru **b** cel mai mic divizor propriu al numărului **a**, iar prin cel de-al treilea parametru **c** furnizează cel mai mare divizor propriu al numărului **a**. Un divizor propriu al unui număr natural nenul **n** este un divizor al său diferit de **1** și **n**. Dacă numărul **n** nu are nici un divizor propriu, atunci parametrii **b** și **c** trebuie să furnizeze amândoi valoarea **0**. Dacă numărul **n** are un singur divizor propriu **d**, atunci parametrii **b** și **c** trebuie să fie furnizeze amândoi valoarea lui **d**. **Exemplu:** pentru **a=45**, valoarea returnată prin parametrul **b** va fi **3**, iar valoarea returnată prin **c** va fi **15**. **(10p.)**
2.	Scrieţi un program care citeşte de la tastatură un număr natural **n** (**3≤n≤50**) şi elementele unui tablou bidimensional cu **n** linii şi n coloane, numere naturale cu cel mult **4** cifre, apoi modifică tabloul în memorie, eliminând elementele aflate pe diagonala principală, ca în exemplu. Programul trebuie să afişeze pe ecran tabloul obţinut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeaşi linie fiind separate prin câte un spaţiu. **Exemplu:** pentru **n=4** şi tabloul <pre>7 5 3 4 8 1 5 6 4 6 3 5 0 9 9 6</pre>programul trebuie să afişeze pe ecran tabloul de mai jos: <pre>5 3 4 8 5 6 4 6 5 0 9 9</pre>**(10p.)**
3.	Fişierul text **bac.in** conţine, pe prima linie, cel mult **1000000** de numere naturale nenule, fiecare număr fiind format din cel mult **9** cifre. Oricare două numere consecutive sunt despărţite între ele print câte un spaţiu.

a) Scrieţi un program care, folosind un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină şi scrie în fișierul **`bac.out`** cel mai mic număr natural care se poate obține din cifrele tuturor numerelor din fişierul **`bac.in`**. **(8p.)**

Exemplu: dacă fişierul **`bac.in`** conţine numerele **`2117 90 885 515 37`**, atunci fişierul **`bac.out`** trebuie să conţină numărul **`10112355577889`**.

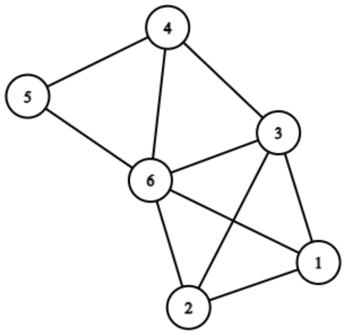
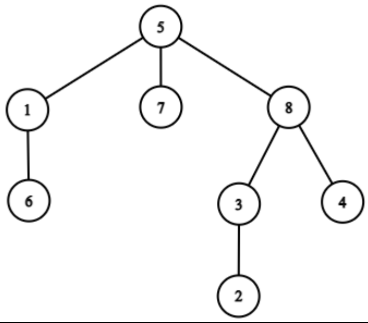
b) Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

⚠ **Observaţii:**

- Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
- Timpul efectiv de lucru este de 3 ore.
- În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
- În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 5 - *Rezolvare*

Subiectul 1

●	Ultima cifră a numărului **n** este egală cu restul împărțirii sale la **10**, iar prima cifră este dată de câtul împărțirii sale la **100**, deci răspunsul corect este **b)**.	
●	Deoarece variabila **q** de tip **Punct3D** conține coordonatele **x** și **y** ale unui punct din spațiu în câmpul **p** de tip **Punct2D**, rezultă că aceste coordonate se pot accesa prin expresiile **q.p.x** și **q.p.y**, în timp ce coordonata **z** poate fi accesată prin expresia **q.z**. În concluzie, răspunsul corect este **c)**.	
●	Folosind metoda backtracking, primele **10** numere formate din **n=5** aflate în ordine strict crescătoare şi având prima cifră cel puțin egală cu **c=3** care se vor genera sunt 34567, 34568, 34569, 34578, 34579, 34589, 34678, 34679, 34689 şi **34789**, deci răspunsul corect este **a)**.	
●	În figura alăturată este reprezentat graful dat în enunț. Gradul minim al unui vârf este **2** (vârful **5**), deci trebuie să eliminăm un număr minim de muchii astfel încât gradele tuturor celorlalte vârfuri să devină egale cu **2**. Se observă faptul că trebuie să eliminăm muchiile (2,3), (6,1), (3,6) și (6,4), deci răspunsul corect este **d)**.	
●	În figura alăturată este reprezentat arborele dat în enunț. Se observă ușor faptul că răspunsul corect este **a)**.	

Subiectul 2

1.	**a)**	Algoritmul calculează câte numere formate doar din cifre pare există între **a** și **b**. Pentru precizarea valorii **4** (între **201** și **208** există **4** numere formate doar din cifre pare: **202**, **204**, **206** și **208**) se acordă **6p.**, iar pentru orice altă valoare se acordă **0p**.
	b)	Deoarece **a=818**, înseamnă că trebuie determinată cea mai mare valoare posibilă pentru **b** astfel încât între **818** și **b** să existe **7** numere formate doar din cifre pare. Se observă faptul că între **818** și **828** există **5** numere cu această proprietate (**820**, **822**, **824**, **826** și **828**), între **829** și **839** nu există nici un număr cu această proprietate (toate conțin cifra impară **3**), iar numerele **840** și **842** au proprietatea cerută. În concluzie, cea mai mare valoare posibilă pentru **b** este **843** (numărul **844** este format doar din cifre pare, deci algoritmul ar afișa valoarea **8**). Pentru precizarea valorii **843** se acordă **6p.**, iar pentru orice altă valoare se acordă **0p**.

c) Structura repetitivă pentru **`i<-a,b execută`** se poate înlocui cu una dintre structurile repetitive **`cât timp... execută`** sau **`repetă... până când`**, având grijă la inițializarea, testarea și incrementarea variabilei **`i`**:

```
citeşte a,b
nr ← 0
i ← a
┌cât timp i<=b execută
│ d ← 1
│ aux ← i
│┌cât timp aux≠0 execută
││ c ← aux%10
││ ┌dacă c%2 = 1 atunci
││ │  d ← 0
││ └■
││ aux ← [aux/10]
│└■
│┌dacă d = 1 atunci
││  nr ← nr+1
│└■
│ i ← i+1
└■
scrie nr
```

```
citeşte a,b
nr ← 0
i ← a
┌repetă
│ d ← 1
│ aux ← i
│┌cât timp aux≠0 execută
││ c ← aux%10
││ ┌dacă c%2 = 1 atunci
││ │  d ← 0
││ └■
││ aux ← [aux/10]
│└■
│┌dacă d = 1 atunci
││  nr ← nr+1
│└■
│ i ← i+1
└până când i>b
scrie nr
```

Atragem atenția asupra faptului că structura repetitivă **`cât timp...execută`** asigură echivalența cu structura **`pentru...execută`** în toate situațiile, pe când structura **`repetă...până când`** este echivalentă doar în unele situații, printre care și cea din acest algoritm!

Pentru alegerea unei structuri repetitive se acordă **2p.**, pentru inițializarea lui **`i`** se acordă **1p.**, pentru testarea condiției de continuare sau oprire se acordă **3p.**, pentru incrementarea lui **`j`** se acordă **1p.**, iar pentru scrierea integrală a restului algoritmului se mai acordă **3p.**, deci se obțin, în total, **10p.**

d) Se testează capacitatea de a reprezenta pe hârtie algoritmul dat folosind un limbaj de programare studiat. Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afișare a rezultatului și de decizie se acordă câte **0.5p.**, pentru cele două instrucțiuni repetitive se acordă **2p.**, iar pentru cele 7 atribuiri se acordă **1p.** Pentru structura corectă a programului se mai acordă **1p.**, deci se obțin, în total, **6p.**

2. În urma executării instrucțiunilor din secvența dată, șirul **`s`** se va transforma astfel: **`"examene"->"exameneame"->"eneame"`**. Pentru scrierea valorii `6` (lungimea șirului **`"eneame"`**) se acordă **6p.**, iar pentru orice altă valoare se acordă **0p**.

3. În urma apelului **`f(100100,1)`** se va obține valoarea **`4`**, deoarece **`f(100100,1)=1+f(10010,1)=1+1+f(1001,1)=1+1+f(100,1)=1+1+1+f(10,1)=1+1+1+1+f(1,1)=1+1+1+1+0=4.`**
Pentru precizarea valorii **`4`**, se acordă **6p.**, iar pentru orice altă valoare se acordă **0p**.

Subiectul 3

<table>
<tr><td>1.</td><td>
Algoritmul caută, mai întâi, cel mai mic divizor propriu d al numărului a. Dacă acesta nu există, atunci parametrii b și c primesc amândoi valoarea 0, altfel parametrul b primește valoarea lui d, iar parametrul c primește valoarea [a/d] (cel mai mare divizor al unui număr se obține împărțind numărul respectiv la cel mai mic divizor al său).

<table>
<tr><th>Limbajul Pascal</th><th>Limbajul C++</th></tr>
<tr><td>

```
procedure divp(a:longint;
               var b,c:longint);
var d:longint;
begin
   d:=2;
   while((d<=a div 2)and
         (a mod d <>0)) do
     d:=d+1;
   if(d>a div 2) then
     begin
       b:=0;
       c:=0;
     end
   else
     begin
       b:=d;
       c:=a div d;
     end;
end;
```

</td><td>

```
void divp(int a, int &b, int &c)
{
   int d;
   d = 2;
   while((d <= a/2)&&(a%d!=0))
     d++;
   if(d>a/2)
     b=c=0;
   else
   {
      b = d;
      c = a/b;
   }
}
```

</td></tr>
</table>

Pentru respectarea structurii antetului (procedure/void) se acordă 1p., pentru declararea corectă a parametrului de intrare a încă 1p., pentru declararea corectă a parametrilor de ieșire b și c încă 1p., pentru declararea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p. Corectitudinea algoritmică a prelucrării în vederea obținerii valorii cerute este notată cu 4p. Pentru transmiterea naturală a rezultatului prin parametrii b și c se acordă 1p, aceasta însemnând că valorile lui b și c nu se afișează sau nu se returnează explicit. Astfel, în total, se obțin 10p.
</td></tr>
<tr><td>2.</td><td>
Pentru a elimina elementul aflat pe diagonala principală pe o linie i a matricei (respectiv elementul aflat pe coloana j=i), se vor deplasa cu o poziție spre stânga elementele aflate pe următoarele poziții (respectiv pe coloanele i+1,i+2,...,n), obținându-se astfel o matrice cu n linii și n-1 coloane (am considerat faptul că liniile și coloane matricei sunt numerotate de la 0) :

```
┌pentru i ← 0,n-1 execută
│┌pentru j ← i,n-2 execută
││  a[i][j] ← a[i][j+1]
│└■
└■
```

Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui n se mai acordă 1p., pentru citirea elementelor matricei se mai acordă 1p., pentru eliminarea corectă a tuturor elementelor aflate pe diagonala principală se mai acordă 4p., iar pentru
</td></tr>
</table>

	afișarea matricei conform cerinței se acordă **2p.** În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă **1p.**, deci se obțin, în total, **10p.**	
3.	**b)**	Vom utiliza un vector de frecvențe **`fc`** cu **`10`** elemente pentru a contoriza aparițiile fiecărei cifre. Vom parcurge fișierul **`bac.in`** caracter cu caracter, folosind variabila **`c`** de tip **`char`**, și, în cazul în care variabila **`c`** conține o cifră, îi vom crește frecvența sa cu **`1`**. Ținând cont de faptul că un număr nu poate să înceapă cu cifra **`0`**, vom determina cel mai mic număr care se poate forma folosind cifrele tuturor numerelor din fișierul **`bac.in`** astfel: • căutăm prima cifră nenulă **`cn`** având frecvența cel puțin egală cu **`1`** (sigur există această cifră deoarece numerele din fișierul **`bac.in`** sunt nenule!), o scriem în fișierul de ieșire **`bac.out`** și îi scădem frecvența cu **`1`**; • începând cu cifra **`0`**, scriem în fișierul de ieșire toate cifrele până la **`9`**, fiecare cifră **`c`** fiind scrisă de un număr de ori egal cu frecvența sa **`fc[c]`**. Eficiența algoritmului din punct de vedere al timpului de executare constă în faptul că parcurgem o singură dată fișierul de intrare **`bac.in`**, obținând astfel un algoritm liniar în raport cu numărul de cifre din fișier. Eficiența algoritmului din punct de vedere al memoriei utilizate constă în faptul că se va folosi doar un vector de frecvențe cu **`10`** componente, evitând astfel memorarea tuturor celor maximum **`9000000`** de cifre din fișier într-un vector de caractere. Pentru o descriere coerentă a metodei se acordă **1p.**, iar pentru justificarea eficienței, încă **1p.**, deci, în total **2p.**
	a)	**Limbajul Pascal**

```pascal
var fin,fout:text;
    c:char;
    i,j:longint;
    cn:byte;
    fc:array[0..9]of byte;
begin
  for i:=0 to 9 do
    fc[i]:=0;

  assign(fin,'bac.in');
  reset(fin);

  while(not eof(fin)) do
    begin
      read(fin,c);
      if((c>='0')and(c<='9')) then
        inc(fc[ord(c)-ord('0')]);
    end;

  close(fin);

  assign(fout,'bac.out');
  rewrite(fout);

  cn:=1;
```

```
  while (fc[cn]=0) do
    cn:=cn+1;

  write(fout,cn);
  fc[cn]:=fc[cn]-1;
  for i:=0 to 9 do
    for j:=1 to fc[i] do
      write(fout,i);

  close(fout);
end.
```

Limbajul C++

```
#include <fstream>

using namespace std;

ifstream fin("bac.in");
ofstream fout("bac.out");

int main()
{
   int fc[10],i,j,cn;
   char c;

   for(i=0;i<10;i++)
     fc[i]=0;

   while(fin>>c)
     if((c>='0')&&(c<='9'))
       fc[c-'0']++;

   fin.close();

   cn=1;
   while(fc[cn]==0)
     cn++;

   fout<<cn;
   fc[cn]--;

   for(i=0;i<=9;i++)
     for(j=0;j<fc[i];j++)
       fout<<i;

   fout.close();
   return 0;
}
```

Se acordă **1p.** pentru operațiile cu fișiere (declarare, nume corect și deschidere pentru citire), încă **1p.** pentru citirea tuturor cifrelor din fișier, **1p.** pentru un algoritm corect, încă **2p.** pentru determinarea numărului cerut și **1p.** pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor etc.).
Se acordă **1p.** pentru alegerea unui algoritm eficient ca timp de executare și **1p.** pentru utilizarea eficientă a memoriei, deci, în total, **8p.**

VARIANTA 6

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Ştiind că iniţial variabilele întregi a şi b au valorile a=10, b=15, indicați care sunt valorile variabilelor a şi e în urma executării instrucţiunii Pascal /C /C++ de mai jos. **(4p.)**

Limbajul Pascal		Limbajul C/C++	
`e := a*b%10 = b%10*a;` `write(a,' ',e);`		`e= a*b%10 == b%10*a;` `cout<<a<<" "<<e; | printf("%d %d \n",a,e);`	
a.	`10 FALSE`	**a.**	`10 0`
b.	`15 FALSE`	**b.**	`15 0`
c.	`5 TRUE`	**c.**	`5 1`
d.	`10 TRUE`	**d.**	`10 1`

2. Numim gradul unui nod din arbore numărul de descendenţi direcţi ai săi. Se dă un arbore care are `1200` noduri de gradul `3`, `300` noduri de gradul `2`, un nod cu gradul `1` şi restul nodurilor de gradul `0`. Care este numărul de muchii pe care le conţine graful? **(4p.)**

a. `4240`	**b.** `4201`	**c.** `4202`	**d.** `4120`

3. Fie G=(X,V) un graf neorientat unde X este mulţimea vârfurilor şi V mulţimea arcelor. Graful G este complet dacă: **(4p.)**

a.	un vârf are cel puţin `n` muchii adiacente
b.	matricea de adiacenţă asociată are toate elementele egale cu `1`
c.	ordinul grafului este `n-1`
d.	are `n(n-1)/2` muchii, unde `n` este numărul de vârfuri

4. Folosind numai cifrele `{0,1,2,3}`, se construiesc, prin metoda Backtracking, toate numerele cu `3` cifre în care oricare două cifre alăturate nu au aceeaşi paritate. Se obţin în ordine numerele: `101`, `103`, `121`, `123`, `210`, ... Utilizând aceeaşi metodă pentru a obţine numere cu patru cifre din multimea `{0,1,2,3,4}` în care oricare două cifre alăturate nu au aceeaşi paritate care este al `4`-lea număr generat? **(4p.)**

a. `1034`	**b.** `1030`	**c.** `1012`	**d.** `1232`

5. Se consideră subprogramul `rec` cu definiţia de mai jos:

Limbajul Pascal	Limbajul C/C++
`procedure rec(n,d:integer);` `begin` `    if(d*d<n) then begin` `        write (d,' ');` `        d:=d+1;` `        rec(n,d+1);` `        write (d,' ');` `      end;` `end;`	`void rec(int n,int d)` `{` `    if(d*d<n)` `    {  cout<<d<<" ";` `       d++;` `       rec(n,d+1);` `       cout<<d<<" | print("%d",d);` `    }` `}`

	Ce se afişează la apelul `rec(17,2)`?			
	a. 2 3 5 4	b. 2 4 5 3	c. 2 4 3 5	d. 2 3 4 5

Subiectul 2 (30p)

Algoritmul alăturat este reprezentat în pseudocod.

1.

```
citeste a,b (numere naturale nenule)
m← a
┌cât timp a>0 sau b>0 execută
| a←b
| citeste b
| ┌dacă a>m atunci
| |  m←a
| └■
| ┌dacă b=0 și a>0 atunci
| | scrie m
| | m←0
| └■
└■
```

a. Scrieţi care sunt valorile afişate dacă se citesc în această ordine valorile: `34,15,24,10, 0,15,49,3,100,8,0,25,2,50,0,0`. **(6p.)**

b. Daţi exemplu de un set de date de intrare pentru care algoritmul afişează aceeaşi valoarea de trei ori. **(6p.)**

c. Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **`cât timp ... execută`** cu o structură repetitivă de alt tip. **(6p.)**

d. Scrieţi programul **Pascal/C/C++** corespunzător algoritmului dat. (10p)

2. Se consideră declarările de mai jos, în care variabila m memorează numele şi data naşterii unui elev. Scrieţi o secvenţă de instrucţiuni care citeşte de la tastatură informţiile despre un elev în variabila **m** şi afişează fie numele elevului dacă anul naşterii este strict mai mic decât 2000, fie data naşterii în caz contrar. **(6p.)**

Limbajul Pascal

```
type data=record
 zi,luna,an:integer;
end;
elev=record
  nume:string[20];
  datan:data;
end;
var m:elev;
```

Limbajul C/C++

```
struct data
{
    int zi,luna,an;
};
struct elev
{
    char nume[20];
    data datan;
};
elev m;
```

3. Variabila **a** memorează şirul de caractere **BacaLauReat**. Care va fi conţinutul variabilei **a** după executarea următoarei secvenţe de instrucţiuni? **(6p.)**

Limbajul Pascal

```
var a:string[30];
    i:integer;
for i:=1 to length(a) do
  if ord(a[i]) in ['A'..'Z'] then
    a[i]:=chr(ord(a[i])+32)
```

Limbajul C/C++

```
char a[31];
int i;
for(i=0;i<strlen(a);i++)
    if(a[i]>='A'&&a[i]<='Z')
    a[i]=a[i]+32;
```

	else delete(a,i,1); write(a);	else {char t[31]; strcpy(t , a + i + 1); strcpy(a + i , t);} cout<<a;|printf("%d",a);

Subiectul 3 (30p)

1.	Scrieţi definiţia completă a subprogramului **p** cu trei parametri, care primeşte prin intermediul parametrilor: - **n,** un număr natural nenul **(n≤100)**; - **x,** un tablou unidimensional cu **n** componente întregi de maximum patru cifre fiecare, indexat de la 0; - **k**, un număr natural nenul **(k≤100)**. Subprogramul returnează poziţia de început a ultimei secvenţe de **k** elemente care conţine cele mai multe numere pare. Dacă toate elementele tabloului unidimensional sunt numere impare se va returna -1. **Exemplu:** pentru **n=10,k=3** şi tabloul **x=(1,2,4,5,6,2,4,6,8,9)** se va returna valoarea **6** iar pentru **n=10,k=3** şi tabloul **x=(1,1,7,5,5,1,3,3,1,9)** se va returna valoarea **-1.** **(10p.)**	
2	Scrieţi un program **Pascal/C/C++** care citeşte de la tastaură un număr natural **(2≤n≤24)** şi **nxm** numere naturale de cel mult patru cifre, elementele unui tablou bidimensional. Programul determină prima respectiv ultima linie cu toate elementele nenule şi interschimbă elementele celor două linii. Programul va afişa matricea pe ecran, câte o linie a tabloului bidimensional pe câte o linie a ecranului, cu câte un spaţiu între elementele fiecărei linii. **(10p.)**	
	Exemplu pentru **n=5, m=5** şi tabloul: 1 2 0 4 3 1 2 3 1 7 1 2 0 7 6 3 2 3 1 1 2 3 4 0 8	Se va afişa: 1 2 0 4 3 3 2 3 1 1 1 2 0 7 6 1 2 3 1 7 2 3 4 0 8
3.	Fişierul text **bac.in** conţine pe prima linie două numere naturale **n** şi **m**, **1≤n≤300, 1≤m≤300** iar pe următoarele linii **n** secvenţe a câte **m** elemente, cifre **0** şi **1.** **a)** Scrieţi un program **Pascal/C/C++** care determină şi afişează pe ecran în mod eficient, din punct de vedere al timpului de execuţie şi al spaţiului de memorie numărul secvenţei şi a locului din cadrul acesteia al fiecărei cifre egale cu **1**. **(8p.)** **b)** Descrieţi succint, în limbaj natural, metoda de rezolvare folosită, explicând în ce constă eficienţa ei. **(2p.)**	
	Exemplu: Dacă fişierul **bac.in** are conţinutul alăturat, atunci pe ecran se va afişa: **1 1;1 3;1 4;3 1;4 2;4 4**	4 6 1 0 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0 0

VARIANTA 6 - *Rezolvare*

Subiectul 1 - rezolvare

1.	În cadrul expresiei se întâlnesc operatori relaţionali şi operatori aritmetici. Se ţine cont de prioritatea operatorilor în evaluarea expresiei (operatorii aritmetici au proritate mai mare decât operatorii relaţionali). Pentru precizarea răspunsului **a)** se acordă **4p**.
2.	Fie **n** numărul de noduri. În acest caz numărul de muchii este **n-1**. Cum **n= 1+1200 x 3 + 300 x 2+ 1 = 4202** rezultă că numărul de muchii este **4201**. Pentru răspunsul **b)** se acordă **(4p.)**
3.	Un graf neorientat este complet dacă oricare două noduri sunt adiacente. Prin urmare un graf este complet dacă numărul total de muchii este **n(n-1)/2.** Pentru alegerea variantei de răspuns **d** se acordă **(6p.)**
4.	Se generează în maniera "înaintare cu revenire", şirurile: **`1010,1012,1014,1030,1032,1034,`** etc. Al patrulea număr generat este `1030`. Pentru răspunsul **b)** se acordă (**4p**.)
5.	La apelul **`rec(17,2)`** deoarece **`2x2<17`** se va afişa valoarea **`2`**, variabila **`d`** va primi valoarea **`3`** şi va avea loc apelul **`rec(17,4).`**Deoarece **`4x4<17`** se va afişa **`4`**, variabila **`d`** va primi valoarea **`5`** şi va avea loc apelul **`rec(17,6).`**Cum **`6x6>17`** nu va mai fi un nou apel şi se vor afişa valorile din memorie în ordinea inversă memorării adică **`5`** şi **`3`**. Pentru răspunsul **`2 4 5 3`** **varianta b** se acordă (4**p**.)

Subiectul 2 - rezolvare

1.	**a)**	Se observă că şirul de valori citite este format din mai multe secvenţe care se termină cu `0`. Parcurgând pas cu pas algoritmul se observă că pentru fiecare secvenţă se determină elementul cu valoarea maximă. Pentru exemplu dat, prima secvenţă are elementul cu valoarea maximă **`34,`** a doua are elementul cu valoarea maximă **`100`**, iar ultima are elementul cu valoarea maximă **`50`**. Deci se vor afişa valorile: **`34,100,50`**. (**6p)**.
	b)	Pentru afişarea aceleiasi valori de trei ori trebuie căutat un set de date de intrare care conţine trei secvenţe finalizate cu 0 care au acelaşi element cu valoare maximă. Un exemplu ar putea fi `33, 4, 5, 0, 7, 11, 33, 0, 23, 33, 0, 0.` Pentru precizarea corectă a unui set de date conform cerinţei se vor acorda **6p.**
	c)	Structura **`cât timp ... execută`** poate fi înlocuită cu structura repetitivă **`repetă ... până când.`**Trebuie să avem în vedere că aceasta se execută cel puţin odată, deci verificăm dacă **`a>0` sau `b>0`**. Prezentăm mai jos o posibilă soluţie:

		citeste a,b (numere naturale nenule) **m← a** **┌dacă a>0 sau b>0 atunci** **| ┌repeta** **| | a←b** **| | citeste b** **| | ┌dacă a>m atunci** **| | |m←a** **| | └■** **| | ┌dacă b=0 și a>0 atunci** **| | | scrie m** **| | | m←0** **| | └■** **| └ până când a<=0 și b<=0** **└■** Pentru integrarea corectă a unei structuri repetitive se acordă **3p**., pentru rescrierea integrală a algoritmului se acordă **3p**. - în total **6p**.
	d)	Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și de decizie se acordă câte **3p**., pentru scrierea corectă a instrucţiunii repetitive se acordă **3p.,** iar pentru atribuiri se acordă **3p.** Pentru structura corectă a programului se mai acordă **1p**., în total **10p**. O posibilă implementare în limbajul de programare studiat este:

Limbajul Pascal	**Limbajul C/C++**
`var a,b,m:integer;` `begin` `   read(a,b);` `   m:=a;` `   while (a>0) or (b>0) do` `    begin` `      a:=b;` `      read(b);` `      if(a>m) then` `        m:=a;` `      if (a>0) and  (b=0) then` `       begin` `         write(m,'  ');` `         m:=a;` `       end;` `    end;` `end.`	`using namespace std;` `int main() {` `int m,a,b;` `cin>>a>>b;` `|scanf("%d%d",&a,&b);` `m=a;` `while(a>0||b>0) {` `        a=b;` `        cin>>b;` `          | scanf("%d",&b);` `        if (a>m) m=a;` `        if ((b==0)&&(a>0))` `        {cout<<m<<"  ";` `           | printf("%d",m);` `          m=0;` `        }` `        }` `        return 0;` `}`

2.	Pentru scrierea corectă a unui set de instrucţiuni şi pentru citirea variabilei **m** se acordă **(3p.)**. Pentru scrierea corectă a instrucţiunii de decizie şi afişarea conţinutului variabilei **m,** conform cerinţei, se acordă încă **(3p.)**. Total **(6p.)** Un exemplu de implementare ar putea fi:

Limbajul Pascal	Limbajul C/C++
<pre>readln(m.nume, m.datan.zi, m.datan.luna, m.datan.an); if(m.datan.an > 2000) then write(m.nume) else write(m.datan.zi, m.datan.luna, m.datan.an);</pre>	<pre>cin>>m.nume>>m.datan.zi >>m.datan.luna>>m.datan.an; |scanf("%s%d%d",&m.nume, &m.datan.zi,&m.datan.luna, &m.datan.an); if(m.datan.an > 2000)cout<<m.nume; |printf("%s",m.nume); else cout<<m.datan.zi <<" "<< m. datan. luna <<" "<<m.datan.an; |printf("%s%d%d",m.nume, m.datan.zi, m.datan.luna, m.datan.an)</pre>

3.	Secvenţa de instrucţiuni menţionată realizează următoarele modificări asupra şirului dat: dacă pe locul **`i`** se află un caracter literă mare acesta devine literă mică altfel caracterul de pe locul **`i`** se şterge. Prin ştergere, caracterul de pe locul **`i+1`** ajunge pe Iocul **i** explicând astfel de ce nu se şterg toate caracterele litere mici şi de ce rămân caractere litere mari neconvertite în litere mici. Prin urmare şirul **`BacaLauReat`** va deveni **`bcLura`.** Pentru rezolvarea corectă a acestei cerinţe se acordă **(6p.)**

Subiectul 3 - rezolvare

1. In cadrul subprogramului se parcurge tabloul unidimensional primit ca parametru, şi se determină numărul elementelor cu valori pare din cadrul unei secvenţe de **k** elemente consecutive. Se returnează poziţia de început a secvenţei cu număr maxim de valori pare sau **0** în cazul în care tabloul nu are elemente cu valori pare.

a) Pentru respectarea structurii antetului **`function`**/**`int`** se acordă **`1p`**., pentru declararea corectă a parametrilor de intrare **`1p`**, pentru respectarea structurii subprogramului şi a sintaxei limbajului se acordă **`1p`**. Pentru determinarea secvenţei cu cele mai multe numere pare se acordă **`2p`.** Pentru transmiterea corectă a rezultatului se acordă **`1p`**. In total **(6p.)**

Un exemplu de implementare poate fi:

Limbajul Pascal	Limbajul C/C++
<pre>function(n:integer;x:vector;k: integer):integer; var i,j,poz,max,s:integer; begin max:=0; poz:=0; for i:=1 to n-k+1 do begin s:=0; for j:=i to k+i-1 do if(x[j] mod 2=0) then inc(s);</pre>	<pre>int p(int n, int x[], int k) { int max1=0,poz=-1,s=0,ok=0; for (int i=0;i<n-k;i++) { for (int j=i;j<=i+k;j++) if (x[j]%2==0) {s++; ok=1;} if (s>=max1) {max1=s; poz=i;} }</pre>

```
        if(s>=max) then
        begin
          max:=s;
          poz:=i;
        end;
        end;
        if(max>0) then
                p:=poz
        else
                p:=0;
end;
```

```
    if(ok)
    return poz;
    else return -1;
}
```

a) Pentru declararea corectă a variabilelor globale se acordă **1p,** pentru citirea valorilor şirului de numere se acordă **1p**. pentru apelarea corectă a subprogramului se acordă **1p.** Pentru corectitudine globală se acordă **1p**.Total (**4p.**)

2. Pentru declararea corectă a variabilelor se acordă **1p.,** pentru citirea variabilelor **n** şi **m** se acordă **1p.,** pentru citirea tabloului bidimensional se acordă **1p.,** pentru determinarea poziţiilor celor două linii care se vor interschimba, **2p**, pentru algoritm corect de interschimbare linii se acordă **2p**., iar pentru afişarea tabloului bidimensional se acordă **1p**. În plus, pentru declararea variabilelor simple, structura şi corectitudinea sintactică a programului se mai acordă **2p**. În total **10p.**

Un exemplu de implementare fi:

Limbajul Pascal

```
var
i,n,j,m,aux,x1,x2:integer;
a:array[1..25,1..25] of
integer;
  vb:boolean;
begin
  read(n,m);
  for i:=1 to n do
        for j:=1 to m do
        i:=1;
        while(i<n) do begin
                vb:=true;
        for j:=1 to m do
        if a[i][j]=0 then
        if(vb=false) then
            inc(i)
        else
         begin
             x1:=i;
             i:=n;
         end;
        end;
        while(i>x1) do begin
        vb:=true;
        for j:=1 to m do
        if (a[i][j]=0) then
         if (vb=false) then
                i:=i-1
                else
                begin
                 x2:=i;
```

Limbajul C/C++

```
#include <iostream>
using namespace std;
int main()
{
int i, j, n,m, v[24][24], p=0, u=0,k;
    cin>>n>>m;
    |scanf("%d %d",&n,&m);
    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            cin>>v[i][j];
|scanf("%d",&v[i][j]);
    for(i=0; i<n; i++)
    {
        k=1;
        for(j=0; j<n && k; j++)
            if(v[i][j]==0)
                k=0;
        if(k)
            if(p==0)
                p=i;
            else u=i;
    }
    for(i=0; i<n; i++)
    {int aux;
        aux=v[p][j];
       v[p][j]=v[u][j];
       v[u][j]=aux;
    }
    for(i=0; i<n; i++)
    {
```

```
                  i:=x1;
                end;
        end;
        for j:=1 to m do
        begin
        aux:=a[x1][j];
        a[x1][j]:=a[x2][j];
        a[x2][j]:=aux;
        end;
        for i:=1 to n do
        begin
        for j:=1 to n do
       write( a[i][j], ' ');
        writeln;
        end;
end.
```

```
                cout<<v[i][j]<<" ";
                |printf("%d",v[i,j]);
            cout<<endl;
        }
        return 0;
}
```

3. a) Se acordă **2p.** pentru operaţia de citire din fişier (declarare, nume corect şi deschidere pentru citire), **2p**. pentru citirea tuturor numerelor din fişier, **2p.** pentru un algoritm corect de afişare a valorilor cerute, **2p.** pentru corectitudinea globală (declararea variabilelor, structura programului, sintaxa instrucţiunilor, etc.). Total **8** puncte.

Un exemplu de implementare poate fi:

Limbajul Pascal

```
var
  f:text;
  n,i,m,j,x:integer;
begin
   assign(f,'date.in');
   reset(f);
   readln(f,n,m);
   for i:=1 to n do
        for j:=1 to m do
        begin
                read(f,x);
                if(x=1) then
                write(i,'
',j,';')
        end;
        close(f);
        readln;
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
|FILE *f=fopen("bac.txt","r");
int main()
{
  int n,i,j,m,x;
  f>>n>>m;
     |fscanf(f,"%d%d",&n, &m);
        for(i=1;i<=n;i++)
          for(j=1;j<=m;j++)
          {
           f>>x;
          |fscanf(f,"%d",&v[i]);
          if(x==1)
           cout<<i<<" "<<j<<";";
           }
        f.close();
        return 0;
}
```

b) Algoritmul este eficient deoarece pentru implementare nu se utilizează tablouri unidimensionale sau bidimensionale iar parcurgerea elementelor din fişier se face o singură dată. Se parcurge fişierul `date.in` şi se afişează coordonatele fiecărui element nenul (secvenţa:**i;** locul în cadrul secvenţei **:j,**unde **i** parcurge toate secvenţele **`i=1,n`** iar **`j`** parcurge toate elementele unei secvenţe **`j=1,m`**). Pentru o descriere coerentă a metodei se acordă **`1p.`**, iar pentru justificarea eficienţei, încă **`1p.`** - în total **`2p.`**

VARIANTA 7

SUBIECTUL I (20 de puncte)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Variabila **x** este de tip întreg şi poate memora un număr întreg cu cel mult o cifră. Indicaţi valoarea minimă pe care o poate avea expresia alăturată.

Limbajul Pascal	Limbajul C/C++
`x mod 5 - 1`	`x % 5 - 1`

a. `-5` **b.** `-1` **c.** 3 **d.** `4`

2. Fie subprogramul `f` definit mai jos. Indicaţi valoarea `f(315)`.

Limbajul Pascal

```
function f(x:integer):integer;
begin
      if x mod 10 = 0 then f:=0
      else
            if x mod 3= 0 then
                  f:=f(x-2)
            else
                  f:=f(x div 10) + 1
end;
```

Limbajul C/C++

```
int f(int x)
{
      if(x%10==0) return 0;
      else
            if(x%3==0)
                  return f(x-2);
      return f(x/10)+1;
}
```

a. 3 **b.** 6 **c.** 7 **d.** 9

3. Utilizând metoda backtracking, se generează toate posibilităţile de a obţine suma `6` cu numere naturale nenule distincte. Două sume sunt distincte dacă diferă prin cel puţin un termen. Soluţiile generate sunt, în această ordine, `1+2+3`, `1+5`, `2+4`. Aplicând acelaşi algoritm pentru a genera toate posibilităţile de a obţine suma `10`, dacă prima soluţie generată este `1+2+3+4`, indicaţi a câta soluţie este soluţia `2+8`.

a. `a 5-a` **b.** `a 6-a` **c.** `a 7-a` **d.** `a 8-a`

4. Un arbore cu rădăcină are `5` noduri, numerotate de la `1` la `5`. Indicaţi şirul de valori care nu poate fi vector de "taţi" pentru arbore.

a. `(2,0,1,1,2)` **b.** `(5,5,1,1,0)` **c.** `(5,1,0,2,3)` **d.** `(0,4,1,2,1)`

5. Se consideră un graf neorientat complet, cu `5` noduri. Indicaţi numărul maxim de muchii ce pot fi eliminate pentru a obţine un graf parţial al său cu trei componente conexe.

a. 6 **b.** 7 **c.** 8 **d.** 9

SUBIECTUL al II-lea (40 de puncte)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Algoritmul alăturat este reprezentat în pseudocod.

S-a notat cu **a%b** restul împărţirii numărului natural **a** la numărul natural nenul **b** şi cu **[c]** partea întreagă a numărului real **c**.

```
citeşte a,b (numere naturale nenule)
p←1
c←0
┌cât timp a≠0 execută
│┌dacă a%b=0 atunci
││ c←c+a%10*p
││ p←p*10
│└■
│ a←[a/10]
└■
scrie c
```

a. Scrieţi ce se afişează dacă se citesc, în această ordine, valorile `12541` şi `3`. **(6p.)**

b. Dacă pentru variabila **b** se citeşte valoarea `9`, scrieţi cel mai mare număr de patru cifre care poate fi citit pentru variabila **a** astfel încât, în urma executării algoritmului, valoarea afişată să fie `0`. **(6p.)**

c. Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

d. Scrieţi în pseudocod un algoritm, echivalent cu cel dat, înlocuind adecvat structura **cât timp...execută** cu o structură repetitivă de alt tip. **(6p.)**

2. În declarările de mai jos, variabila **dn** memorează o dată calendaristică iar variabila **e** memorează în câmpul **d_nt** data naşterii unui elev. Scrieţi o expresie Pascal/C/C++ care să aibă valoarea **true**/1 dacă data corespunzătoare variabilei **dn** coincide cu data naşterii elevului corespunzător variabilei **e** sau valoarea **false**/0 în caz contrar. **(6p.)**

Limbajul Pascal	Limbajul C/C++
`type data=record` `   z,l,a:integer` `end;` `elev=record` `   nume:string;` `   d_nt:data` `end;` `var dn:data;` `    e:elev;`	`struct data {` `   int z,l,a;` `}dn;` `struct elev {` `   char nume[256];` `   data d_nt;` `}e;`

3. Variabilele **s** şi **t** pot memora câtre un șir de cel mult 10 de caractere. Scrieți şirul memorat în variabila **s** în urma executării secvenței alăturate. **(6p.)**

Limbajul Pascal	Limbajul C/C++
`var s:string[10];` `s:='bac2021';` `s[4]:=chr(ord(s[4])-1);` `delete(s,6,2);` `delete(s,1,3);`	`char s[]="bac2021";` `s[3]= s[3]-1;` `strcpy(t,s+7);` `strcpy(s+5,t);` `strcpy(t,s+3);` `strcpy(s,t);`

SUBIECTUL al III-lea **(30 de puncte)**

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Subprogramul **pare** are patru parametri**:**
- **n, a** şi **b** prin care primeşte câte un număr natural ($n<10^3$);
- **v**, prin care primeşte un tablou unidimensional cu **n** elemente, numere naturale nenule, de cel mult trei cifre fiecare;

Subprogramul determină şi returnează numărul de valori pare din intervalul **[a;b]** care aparţin tabloului **v**;
Scrieți definiția completă a subprogramului.
Exemplu: pentru **n=7**, tabloul (15, <u>20</u>, 3, 12, 5, 45, <u>24</u>), **a=14**, **b=30** se returnează 2. **(10p.)**

2. Scrieţi un program Pascal/C/C++ care citeşte de la tastatură două numere naturale, **m** şi **n** (2≤**m**≤20, 2≤**n**≤20), şi construieşte în memorie un tablou bidimensional **A**, cu **m** linii şi **n** coloane, astfel încât parcurgându-l linie cu linie, de sus în jos,şi fiecare linie de la stânga la dreapta,să se obţină şirul cifrelor impare, ordonat crescător. Programul afişează pe ecran tabloul obţinut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeaşi linie fiind separate prin câte un spaţiu.
Exemplu, pentru **m=5** şi **n=4** se obţine tabloul următor:
Programul afişează pe ecran tabloul construit, fiecare linie a tabloului pe câte o linie a ecranului, cu elementele aflate pe aceeaşi linie separate prin câte un spaţiu.
Exemplu: dacă pentru **m=5** şi **n=4** se afişează pe ecran tabloul alăturat.

```
1 3 5 7
9 1 3 5
7 9 1 3
5 7 9 1
3 5 7 9
```

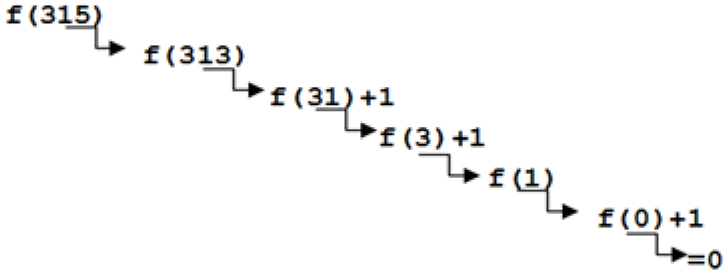
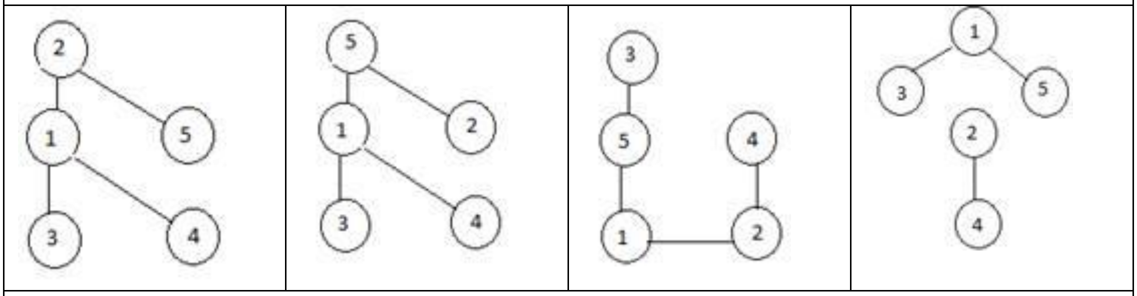
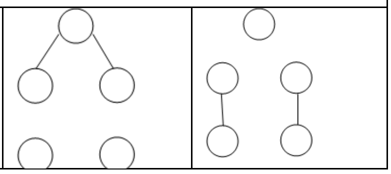
3. Fişierul **bac.in** conţine pe prima linie un număr natural par **x** (0≤**x**≤9) şi pe următoarea linie cel mult 1000000 de numere naturale cu exact o cifră.
Se cere să se afişeze pe ecran numărul de valori impare care se află între prima şi ultima apariţie a lui **x** în fişier. Dacă **x** nu apare de cel puţin două ori se va afişa mesajul **nu exista**.
Proiectați un algoritm eficient din punctul de vedere al memoriei utilizate şi al timpului de executare.
Exemplu: dacă fişierul conţine numerele 2
3 1 <u>2</u> 4 <u>1</u> <u>3</u> <u>5</u> 2 4 <u>1</u> <u>3</u> <u>5</u> 2 0 4 <u>7</u> <u>9</u> <u>5</u> <u>2</u> 4 7 8
pe ecran se afişează 9

a. Scrieți programul Pascal/C/C++ corespunzător algoritmului proiectat. **(8p.)**

b. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. **(2p.)**

VARIANTA 7 - *Rezolvare*

SUBIECTUL I

1.	Rezultatul expresiei **`x mod 5`** (Pascal) sau **`x%5`** (C/C++) este negativ în cazul unui operand **`x`** întreg negativ. Pentru precizarea răspunsului **a)** se acordă **4p**.
2.	Se urmăresc apelurile recursive şi se fac calculele:  **`f(315)=3`** Pentru precizarea răspunsului **a)** se acordă **4p**.
3.	Se generează în maniera "înaintare cu revenire", sumele: **`1+2+3+4`**, **`1+2+7`**, **`1+3+6`**, **`1+4+5`**, **`1+9`**, **`2+3+5`**, **`2+8`**,..., etc. Pentru precizarea răspunsului **c)** se acordă **4p**.
4.	Dacă reprezentăm grafic fiecare "arbore" vom obţine:  Se observă că ultimul graf nu poate fi arbore întrucât are două componente conexe. Pentru precizarea răspunsului **d)** se acordă **4p**.
5.	Graful complet cu **`5`** noduri are **`10`** muchii. Pentru a obţine trei componente conexe, numărul maxim de muchii care pot fi eliminate este **`8`**. Un asemenea graf partial ar putea fi reprezentat graphic ca în figurile următpare. Pentru precizarea răspunsului **c)** se acordă **4p**. 

SUBIECTUL al II-lea

1.	**a.**	Se formează un număr **`c`** din ultima cifră a fiecărui prefix divizibil cu **`b`** al lui **`a`**. Pentru precizarea valorii **`24`** se acordă **6p**.
	b.	Se are în vedere că numărul căutat nu trebuie să fie format din prefixe divizibile cu **`9`**. Pentru precizarea valorii **`8999`**, se acordă **6p**.
	c.	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deşi scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerinţă se valorifică experienţele anterioare de implementare şi testare a programelor pe calculator, experienţe care au cimentat cunoştinţele privind structura programelor, declararea variabilelor, sintaxa instrucţiunilor programului, regulile de scriere a expresiilor, etc. Pentru instrucţiunile corecte de declarare a variabilelor se acordă 1p., pentru citirea datelor se acordă 1p., pentru afişarea rezultatului se acordă 1p., pentru instrucţiunea de decizie se acordă 2p., pentru instrucţiunea repetitivă se acordă 2p., iar pentru cele 5 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.

<table>
<tr><td></td><td>d.</td><td>Deoarece valoarea lui a este nenulă, se poate înlocui pur şi simplu structura cât timp a≠0 execută cu o structură repetă...până când. Evident, la înlocuire, se va modifica testul final:
Pentru alegerea unei structuri repetitive se acordă 2p., pentru testarea condiţiei de oprire se acordă 2p., iar pentru scrierea integrală a algoritmului se mai acordă 2p. - în total 6p.</td><td><pre>citeşte a,b (numere naturale nenule)
p←1
c←0
┌repetă
│┌dacă a%b=0 atunci
││ c←c+a%10*p
││ p←p*10
│└■
│ a←[a/10]
└până când a=0
scrie c</pre></td></tr>
<tr><td>2.</td><td colspan="3">Pentru ca datele să coincidă trebuie să aibă aceleaşi valori pentru zi, lună şi an. Dacă e este variabila de tip elev atunci data naşterii elevului este e.d_nt.z,e.d_nt.l, e.d_nt.a, iar data corespunzătoare variabilei dn este dn.z, dn.l, dn.a.
Limbajul Pascal
(e.d_nt.z=dn.z) and (e.d_nt.l=dn.l) and (e.d_nt.a=dn.a)
Limbajul C/C++
e.d_nt.z==dn.z&&e.d_nt.l==dn.l&&e.d_nt.a==dn.a
Pentru acces la câmpurile variabile dn se acordă 1p., pentru acces la câmpurile variabilei e se acordă 2p., iar pentru expresie corectă se mai acordă 3 p. - în total 6p.</td></tr>
<tr><td>3.</td><td colspan="3">În şir, pe poziţia 4 se află caracterul '2 '. Acesta se înlocuieşte cu caracterul '1 '. Se şterg mai întâi ultimele două caractere şi apoi primele trei caractere.
Pentru răspunsul 10 se acordă 6p. (2p. pentru caracterul ‚1', 1p. pentru caracterul '0 ' şi încă 3p pentru rezultat integral corect).</td></tr>
</table>

SUBIECTUL al III-lea

<table>
<tr><td>1.</td><td>Prelucrarea oferită de subprogram parcurge tabloul de la primul element la ultimul şi verifică dacă elementele sunt pare şi în intervalul [a,b].</td></tr>
</table>

Limbajul Pascal	Limbajul C/C++
see below	see below

```pascal
type vector=array[1..100] of integer;
function pare(n,a,b:integer;v:vector):integer;
var i,k:integer;
begin
   k:=0;
   for i:=1 to n do
      if (v[i] mod 2=0) and (v[i]>=a) and (v[i]<=b)
         then k:=k+1;
   pare:=k
end;
```

```cpp
int pare(int n, int v[], int a, int b)
{
   int i,k=0;
   for(i=0;i<n;i++)
      if(v[i]%2==0 && v[i]>=a && v[i]<=b)
         k++;
   return k;
}
```

Pentru respectarea structurii antetului (**function/int**) se acordă 1p., pentru declararea corectă a parametrilor de intrare **n**, **a** şi **b** încă 1p., pentru declararea corectă a parametrului **v** încă 1p., pentru declararea şi iniţializarea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului şi a sintaxei limbajului se acordă încă 1p., pentru rezultat încă 2p. Corectitudinea algoritmică a prelucrării în vederea obţinerii valorii cerute este notată cu 3p. (paritate, apartenenţă în interval, incrementare variabilă locală). În total **10p.**

Facem câteva observaţii: subprogramul **trebuie** să se numească **pare**; În Pascal, parametrul **v** poate fi transmis prin adresă (**var v:vector**), deoarece valorile din tablou nu se modifică şi este mai eficient ca timp de executare.

<table>
<tr><td>2.</td><td>Valorile cu care se completează tabloul bidimensional sunt cifrele impare (1, 3, 5, 7, 9).
Trebuie ţinut cont de faptul că odată ce valoarea unui element depăşeşte 9, aceasta se resetează la 1.
Liniile şi coloanele matricei se numerotează de la 1 (Pascal) sau de la 0 (C/C++).
Soluţii posibile:</td></tr>
</table>

```
val←1
┌pentru i←1,m execută
|     ┌pentru j←1,n execută
|     |     ┌dacă val>9 atunci
|     |     |     val←1
|     |     └■
|     |     a[i,j]←val
|     |     val←val+2
|     └■
└■
```

Sau se completează tabloul cu ultima cifră a valorii memorate în variabila **val**:

```
val←1
┌pentru i←1,m execută
|     ┌pentru j←1,n execută
|     |     a[i,j]←val%10
|     |     val←val+2
|     └■
└■
```

Pentru declararea corectă a tabloului se acordă 1p., pentru citirea lui **m** şi **n** se mai acordă 1p., pentru accesul la un element din tablou se acordă 1p., pentru parcurgerea tabloului pe linii şi coloane se acordă 1p., pentru completarea corectă a tuturor elementelor tabloului se acordă 3p., iar pentru afişarea matricei conform cerinţei se acordă 2p. În plus, pentru declararea variabilelor simple, structura şi corectitudinea sintactică a programului se mai acordă 1p. - în total **10p**.

3. **a.**

Limbajul Pascal

```pascal
Var   f:text;
   x, k, val,nr, nr_imp:longint;
   gasit1,gasit2:boolean;
begin
   assign(f,'bac.in'); reset(f);
   readln(f,x);
   while not eof(f) do
   begin
      read(f,val);
      if val mod 2 =1 then inc(k);
      if val=x then
         if gasit1=false then
         begin
            gasit1:=true;
            k:=0
         end
         else
         begin
            nr_imp:=nr_imp+k;
            k:=0;
            gasit2:=true
         end;
   end;
   close(f);
   if gasit2=false then
      write('nu exista')
   else
      write(nr_imp)
end.
```

Limbajul C/C++

```cpp
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int x,gasit1,gasit2,k,val,nr_imp;
int main()
{
   f>>x;
   while(f>>val)
   {
      if(val%2)k++;
      if(val==x)
      {
         if(!gasit1)
         {
            gasit1=1;
            k=0;
         }
         else
         {
            nr_imp+=k;
            k=0;
            gasit2=1;
         }
      }
   }
   f.close();
   if(!gasit2)
      cout<<"nu exista";
   else
      cout<<nr_imp;
   return 0;
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect şi deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principal corect, încă 1p. pentru determinarea tuturor valorilor cerute, 1p. pentru

		afişarea corectă a rezultatului (mesaj / valoare numerică) şi 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucţiunilor, etc.). Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare (`O(n)`) şi 1p. pentru utilizarea eficentă a memoriei. În total **8p.** Menţionăm că prin algoritm principial corect se înţelege un algoritm care, în intenţie, urmăreşte să rezolve corect problema, deşi poate să prezinte scăpări logice sau confuzii.
	b.	O soluţie posibilă: Parcurgem şirul şi numărăm (într-o variabilă **`k`**) valorile impare dintr-o secvenţă delimitată de două valori **`x`** consecutive. La sfârşitul fiecărei astfel de secvenţe (cu excepţia primeia), vom adăuga numărul **`k`** la valoarea cerută în problemă. Variabilele **`gasit1`** şi **`gasit2`** ne ajută să semnalizăm apariţia primului **`x`** din şir şi respectiv a altui **`x`** din şir afară de primul. Eficienţa algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată şirul cu un număr impresionant de numere. Deoarece numărul de valori din fişier (**`1000000`**) este dimensiunea care contează în problemă, spunem că am obţinut un algoritm liniar. Ca spaţiu de memorie, soluţia propusă este eficientă, deoarece va utiliza doar variabile simple de lucru. Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienţei, încă 1p. - în total **2p.** Menţionăm că descrierea coerentă a problemei nu presupune să povestim programul, ci ideea de prelucrare.

VARIANTA 8

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Indicaţi care este valoarea expresiei **Pascal/C/C++** următoare dacă variabila **m** are valoarea **1234**.

Limbajul Pascal	Limbajul C/C++
`m:=m-m div 10*10`	`m=m-m/10*10`

a. 2	b. 4	c. 3	d. 1

2. Se consideră declarările de mai jos, în care variabila **m** memorează datele unui tablou unidimensional ale cărui elemente reţin denumirea şi preţul unor materiale. Alegeţi varianta corectă care accesează prima literă a **denumirii** celui de-al **treilea** element al tabloului unidimensional.

Limbajul Pascal

```
type material =record
      denumire:string[20];
      pret:real;
 end;
var m:array [1..20] of material;
```

Limbajul C/C++

```
struct material
{   char denumire[30];
    float pret;
} m[20];
```

	Limbajul Pascal	Limbajul C/C++
a	`m[3].denumire[3]`	`m[3].denumire[3]`
b	`m[3].denumire[1]`	`m[2].denumire[0]`
c	`material.denumire[1]`	`material.denumire[0]`
d	`m[4].denumire[1]`	`m[3].denumire[0]`

3. Utilizând metoda backtracking se generează toate şirurile formate din cifrele binare 0 şi 1. Astfel, se generează în această ordine pentru **n=3** şirurile: `111, 110,101, 100,011,010, 001,000`. Pentru **n=4** care vor fi cele două şiruri afişate imediat după şirul `1001`?

a. 0111 1000	b. 1000 1100	c. 0011 1000	d. 1000 1101

4. Se consideră un graf neorientat cu nodurile **1, 2, 3, 4, 5, 6, 7, 8** şi muchiile **[1,2] [1,5], [2,8], [3,7], [4,5], [5,7], [6,4], [7,6], [8,3], [8,7]**. Care este numărul minim de muchii ce pot fi eliminate astfel încât graful obţinut să aibă **trei** componente conexe?

a. 1	b. 2	c. 3	d. 4

5.	Se consideră arborele cu rădăcină având 9 noduri numerotate de la 1 la 9, dat prin vectorul "de taţi" de mai jos: T=(5,5,2,6,0,5,6,1,3) Specificaţi câte noduri pot fi alese ca rădăcină astfel încât numărul nodurilor terminale să fie minim.			
	a. 4	b. 3	c. 2	d. 1

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	**Se consideră algoritmul următor reprezentat în pseudocod:** S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte n,a,b
         (numere naturale nenule)
┌dacă a>b atunci
|b←b+a; a←b-a; b←b-a
└■
┌cât timp n>9 execută
|  s←0
| ┌cât timp n>0 execută
| | s←s+n%10
| | n←[n/10]
| └■
|  n←s
└■
┌cât timp a % n≠0 execută
|  a← a+1
└■
┌dacă a≤b atunci
|  scrie 1
| altfel
|  scrie 0
└■
```

a)	Scrieţi care este valoarea afişată dacă se citesc în această ordine valorile: 33, 30, 10? **(6p.)**
b)	Dacă **n=99** şi **a=55**, stabiliţi cea mai mică valoare care se poate citi pentru variabila **b**, astfel încât, după executarea algoritmului de mai sus, să se afişeze valoarea 0. **(6p.)**
c)	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască ultima structură **cât timp...execută** cu o structură liniară. **(6p.)**
d)	Scrieţi programul **Pascal/C/C++** corespunzător algoritmului dat. **(10p.)**

2. Se consideră următoarea secvenţă de program în care variabila **a** memorează elementele unui tablou bidimensional cu **5** linii şi **5** coloane, numerotate de la **1** la **5**, iar celelalte variabile sunt de tip întreg.

Limbajul Pascal	Limbajul C/C++
`var a:arry[1..5,1..5] of integer;` `    aux:integer;` `for i:=1 to 5 div 2 do` ` for j:=i to 6-i do` `  begin` `   aux:=a[i,j];` `   a[i,j]:=a[6-i,j];` `   a[6-i,j]:=aux;` `end;`	`int a[6][6];` `for(i=1;i<=2;i++)` `  for(j=i;j<=6-i;j++)` `  {` `     int aux;` `     aux=a[i][j];` `     a[i][j]=a[6-i][j];` `     a[6-i][j]=aux;` `  }`

Specificaţi care va fi conţinutul variabilei **a** în urma executării secvenţei de mai sus dacă iniţial variabila **a** avea conţinutul de mai jos: **(6p.)**

```
1 1 1 1 1
2 2 2 2 2
3 3 3 3 3
4 4 4 4 4
5 5 5 5 5
```

3. Se consideră subprogramul **f** cu definiţia de mai jos: **(6p.)**

Limbajul Pascal	Limbajul C/C++
`function f(a,b,c:integer):integer;` `var m:integer;` `begin` `   if(a<=b) then` `    begin` `    m:=(a+b) div 2;` `    if(c<m) then f:=f(a,m-1,c)` `    else` `    if(c>m) then f:=f(m+1,b,c)` `        else` `           f:=b+a;` `       end` `    else  f:=b-a;` `end;`	`int f(int a,int b, int c)` `{` ` if(a<=b)` ` {` `   int m=(a+b)/2;` `   if(c<m)` `      return f(a,m-1,c);` `   else` `     if(c>m)` `       return f(m+1,b,c);` `     else` `       return b+a;` ` }` ` else` `         return b-a;` `}`

a. Ce se afişează la apelul `f(2,7,5)`? **(3p.)**

b. Pentru **a=4** şi **b=9** scrieţi o valoare pentru parametrul **c** astfel încât la apelul `f(a,b,c)` să se afişeze valoarea `10`. **(3p.)**

Subiectul 3 (30p)

	Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.
1.	Subprogramul **suma_x** are trei parametri, prin care primește: • **n**, un număr natural **(1≤n≤100)**; • **a**, un tablou unidimensional cu **n** valori naturale; • **s**, un număr natural din intervalul [1, 1000]. Subprogramul returnează cea mai lungă secvenţă de elemente consecutive care au suma **s .** Dacă nu există o astfel de secvenţă subprogramul returnează valoarea -1. **Exemplu**: Dacă **n=8**, **a=(2,20,-10,5,2,3,1,2)**, **s=10**, subprogramul returnează valoarea 3. **(10p.)**
2.	Se consideră un text cu maximum **250** de caractere, format din litere mici ale alfabetului englez şi spaţii. Cuvintele sunt separate printr-un singur spaţiu. Scrieţi programul **Pascal/C/C++** care citeşte un text ca cel menţionat şi afişează pe ecran acele cuvinte în care vocalele apar în ordine alfabetică (a,e,i,o,u) şi nicio vocală nu se repetă. **Exemplu:** Pentru textul **vaporul pluteşte pe mare** se va afişa **vaporul pe mare** **(10p.)**
3.	Fişierul de intrare **date.in** conţine un şir cu cel mult un milion de numere naturale cu maximum nouă cifre fiecare, separate prin câte un spaţiu. a) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare, citeşte din fişier toţi termenii şirului, determină şi afişează în fişierul **date.out**, numărul termenilor care încep şi se termină cu cifra **1**, apoi numărul termenilor care încep şi se termină cu cifra **2,...,** numărul termenilor care încep şi se termină cu cifra **9**. Cele nouă numere determinate se vor afişa pe prima linie a fişierului, separate prin câte un spaţiu. **(8p.)** Exemplu: Dacă fişierul **date.in** are conţinutul: **13 2842 121 9839 94 9 223 22 1 66 454 43 33 383** Fişierul **date.out** va avea conţinutul: **2 2 2 1 0 1 0 0 2** deoarece sunt doi termeni care încep şi se termină cu **1 (121, 1)** doi termeni care încep şi se termină cu **2 (2842,22),** etc. b) Descrieţi în limbaj natural metoda de rezolvare folosită şi explicaţi în ce constă eficienţa ei. **(2p.)**

VARIANTA 8 - *Rezolvare*

Subiectul 1

1.	Se ţine cont de prioritatea operatorilor în evaluarea expresiei. Pentru raspunsul `b)` se acordă **4p**.
2.	Accesarea denumirii celui de-al treilea element al tabloului unidimensional este **`m[2].denumire`** (C++/C), **`m[3].denumire`** (Pascal), iar pentru accesarea primei litere utilizăm notaţia **`m[2].denumire[0]`** (C++/C) / **`m[3].denumire[1]`** (Pascal). Pentru răspunsul **b)** se acordă **4p.**
3.	Se generează şirurile: **`1111`**, **`1110`**, **`1101`**, **`1100`**, **`1011`** etc. După șirul **`1001`** se află şirurile **`1000,0111`**. Pentru răspunsul **a)** se acordă **4p**.
4.	Numărul minim de muchii ce se pot elimina este `3`. Pentru precizarea răspunsului `c)` se acordă **`4p`**.
5.	Pentru ca numărul nodurilor terminale să fie minim trebuie ales ca rădăcină unul din nodurile terminale actuale. Deci oricare din nodurile `8, 9, 4` sau `7` dacă sunt alese ca rădăcină arborele va avea numai `3` noduri terminale. Pentru precizarea răspunsului `a)` se acordă **`4p`**.

Subiectul 2

1.	**a)**	Numim cifra de control a unui număr cifra obţinută prin adunarea repetată a cifrelor sale până cînd numărul obţinut este mai mic decât **`10`**. Exemplu: cifra de control a numărului **`557`** este **`8`** deoarece **`5+5+7=17`**. Cum **17**>**10** adunăm cifrele numărului **17**, **1**+**7**=**8**, şi **8**<**10** deci cifra de control a numărului **`557`** este **`8`**. Algoritmul verifică dacă un multiplu al cifrei de control corespunzător numărului **n** se găseşte în intervalul **`[a,b]`**. În caz afirmativ se afişează **`1`**. Pentru datele de intrare **`33, 30, 10`** se verifică dacă un multiplu al cifrei de control pentru numărul **`33`** (care este **`6`**) se află în intervalul **`[10,30].`** Cum **`12`** aparţine intervalului **`[10,30],`** se va afişa **`1`**, răspuns pentru care se acordă (**6p**.)
	b)	Pentru afişarea valorii `0`, în cazul în care **`n=99`** şi **`a=55`** trebuie căutată o valoare pentru **`b`** astfel încât nici un multiplu al cifrei de control să nu aparţină intervalului [**`a,b].`** Prin urmare cea mai mică valoare a lui b este **55**. Pentru precizarea corectă a valorii variabilei **`b`** se vor acorda (**6p**.)
	c)	Ultima structură **`cât timp ... execută`** poate fi înlocuită cu structura liniară: **`a<-([a/n]+1)*n`** (**6p**.)
	d)	Pentru declararea variabilelor se acordă **`1p`**, pentru citirea datelor se acordă **`1p`**, pentru afişarea rezultatului se acordă **`1p,`** pentru instrucţiunea de decizie se acordă **`1p`**., pentru cele trei instrucţiuni repetitive se acordă **`3p.,`** iar pentru instrucţiuni de atribuire se acordă **2p.** Pentru structura corectă a programului se acordă **1p**., în total **10p**.

O posibilă implementare în limbajul de programare studiat este:

Limbajul Pascal

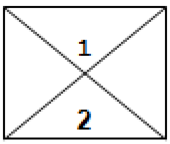
```
var  n,a,b,s:integer;
begin
   read(n,a,b);
   if(a>b) then
        begin
          b:=b+a;
          a:=b-a;
          b:=b-a;
        end;
while (n>9) do
begin
   s:=0;
   while (n>0) do
        begin
          s:=s+n mod 10;
          n:=n div 10;
        end;
      n:=s;
   end;
while (a<=b) and (a mod n<>0)
do
    a:=a+1;
  if(a<=b) then write('1')
        else write('0');
end.
```

Limbajul C/C++

```
#include <iostream>
|#include <stdio.h>
using namespace std;
int main()
{
    int n,a,b;
    cin>>n>>a>>b;
   |scanf("%d %d %d",n,a,b);
         if(a>b)
         {       b=b+a;
                 a=b-a;
                 b=b-a;
         }
         while(n>9)
         {  int s=0;
            while(n)
                 {s=s+n%10;
                  n=n/10;
                 }
            n=s;
         }
         while (a%n!=0)
                 a++;
         if(a<=b) cout << 1;
         else cout<<0;
}
```

2.

Secvenţa interschimbă elementele triunghiurilor **unu** şi **doi** inclusiv elementele de pe diagonale (vezi desenul de mai jos) din tabloul bidimensional, celelalte elemente rămânând neschimbate:




```
5 5 5 5 5
2 4 4 4 2
3 3 3 3 3
4 2 2 2 4
1 1 1 1 1
```

Pentru răspuns corect se acordă (**6p.**).

Pentru răspuns parţial corect (interschimbă doar elementele celor două diagonale sau numai elementele dintre diagonale) se acordă doar (**3p.**)

3.

a) La apelul **`f(2,7,5)`** variabila **`m`** primeşte valoarea **`4`** şi cum **`c>4`** se execută apelul recursiv **`f(5,7,5)`**. La acest apel variabila **`m`** primeşte valoarea **`6`**. Cum **`c<6`** noul apel recursive va fi **`f(5,5,5)`**. La acest apel **`m=c`** deci se va returna valoarea `10`. Pentru răspunsul `10` se acordă (**3p.**)

	b) Dacă **a=4** şi **b=9** variabila **m** primeşte valoarea **6**. În acest caz variabila **c** trebuie să fie mai mică decât **6** sau mai mare decât **6** (în caz contrar se va returna **13**). Dacă **c>6** va avea loc apelul recursiv **f(7,9,c)** caz în care se va returna o valoare>10**.** Prin urmare **c<6**. Dacă **c<5** atunci se va returna o valoare mai mică decât **10** deci **c=5**, răspuns pentru care se primeşte (**3p.**)

Subiectul 3

1. O posibila implementare a subprogramului este:

Limbajul Pascal	Limbajul C/C++

```
function suma
(n,s:integer;a:array[1..100]
of integer):integer;
var i,lmax,lc,ic:integer;
begin
 lmax:=-1;
  for i:=1 to n do
begin
   ic:=i;lc:=0;
  while(lc<>s) and (ic<=n)do
   begin
     lc:=lc+a[ic];inc(ic);
  end;
  if(lc=s) then
     if(ic-i>lmax)then
        lmax:=ic-i;
end;
   suma:=lmax;
end;
```

```
int suma_x(int n,int a[],int s)
{
    int lc,ic,lmax=-1;
    for(int i=0; i<n; i++)
    {ic=i;lc=0;
        while(lc!=s&&ic<n)
        {lc=lc+a[ic];ic++;}
            if(lc==s)
              if(ic-i>lmax)lmax=ic-i;
     }
    return lmax;
}
```

Pentru antet subprogram (se acordă câte **1p**. pentru fiecare aspect al antetului: structură, parametri de intrare, conform cerinței.), **3p**; determinare a valorii cerute **3p**; declarare a tuturor variabilelor locale **1p**; corectitudine globală a subprogramului **3p**. Total **10p.**

2. Pentru declararea corectă a variabilelor se acordă **1p**., pentru citirea corectă a şirului de caractere se acordă **1p**., pentru identificarea cuvintelor în care vocalele apar în ordine alfabetică **4p**., pentru utilizarea corectă a funcţiilor specifice şirurilor de caractere se acordă **2p**., pentru afişarea cuvintelor în care vocalele apar în ordine alfabetică se acordă **1p**. Pentru corectitudinea sintactică a programului se acordă **1p** - în total **10p**.

Un exemplu de implementare fi:

Limbajul Pascal	Limbajul C/C++

```
var
  s:string;
  p:string[20];
  i,k,j:integer;
  vb:boolean;
  vocale:set of char;
begin
  vocale:
=['a','e','i','o','u'];
  readln(s);
```

```
#include <iostream>
#include <string.h>
using namespace std;

int main()
{
    char s[256],*p,v[10];
    cin.get(s,256);
    p=strtok(s," ");
```

		<pre>repeat k:=pos(' ',s); p:=copy(s,1,k-1); vb:=true; if(k=0) then p:=s; for i:=1 to length(p)-1 do for j:=i+1 to length(p) do if(p[i] in vocale) and (p[j] in vocale) and (p[i]>=p[j]) then vb:=false; if(vb) then write(p,' '); delete(s,1,length(p)+1); until k=0; end.</pre> <pre> while(p) { int vb=1; for(int i=0; i<strlen(p)-1 && vb; i++) for(int j=i+1; j<strlen(p) &&vb; j++) if(strchr("aeiou",p[i]) && strchr("aeiou",p[j]) && p[i]>=p[j]) vb=0; if(vb) cout<<p<<" "; p=strtok(NULL," "); } }</pre>
3.	a)	Se acordă **2p.** pentru operaţiile cu fişiere (declarare, nume corect şi deschidere pentru citire şi scriere), **1p**. pentru citirea tuturor numerelor din fişier, **2p.** pentru un algoritm de sortare corect, **1p.** pentru determinarea valorilor cerute, **1p.** pentru afişarea valorilor în ordine crescătoare în fişier **1p.** pentru corectitudinea globală (declararea variabilelor, structura programului, sintaxa instrucţiunilor etc.).**Total 8** puncte. Un exemplu de implementare poate fi:

Limbajul Pascal

```
var
        f,g:text;
        v:array[1..10] of integer;
        i,x,y:integer;
begin
        assign(f,'date.in');
        reset(f);
        assign (g,'date.out');
        rewrite(g);
        while not eof(f) do
           begin
                read(f,x);
                y:=x mod 10;
                while(x>9) do
                  x:=x div 10;
                if(x=y) then
                inc(v[x]);
            end;
        for i:=1 to 9 do
           write(g, v[i], ' ' );
        close(f);
        close(g);
        readln;
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("date.in");
|FILE *f=fopen("date.in"
,"r");
ofstream g("date.out");
int v[10];
int main()
{       int x,i;
        while(f>>x)
        { int y=x%10;
          while(x>9)
             x/=10;
             if(x==y)
             v[x]++;
        }
        for(i=1;i<=9;i++)
        g<<v[i]<<" ";
        g.close();
        f.close();
}
```

	b)	Se parcurge fişierul **`date.in`** şi se reţine într-un tablou unidimensional (vectorul de frecvenţe) numărul elementelor care au prima şi ultima cifră egale. Prin urmare **`v[1]`** reţine numărul elementelor care încep şi se termină cu **1**, **`v[2]`** reţine numărul elementelor care încep şi se termină cu **2** şi aşa mai departe. Se scrie în fişierul **date.out** conținutul tabloului unidimensional. Din punct de vedere al eficienţei de timp, avem o parcurgere liniară a şirului deci **`O(n)`**. Pentru o descriere coerentă a metodei se acordă **1p.,** iar pentru justificarea eficienţei, încă **1p.** - în total **2p**.

VARIANTA 9

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Se dau variabilele `a` şi `b` de tip întreg, `a` memorând valoarea `4`, iar `b` valoarea `2`. Care dintre expresiile `Pascal|C/C++` de mai jos are valoarea `FALSE|0`?

	Limbajul Pascal	Limbajul C/C++
a.	`a-b <> 0`	`a-b!= 0`
b.	`a+b > a mod b+1`	`a+b > a%b+1`
c.	`(a-2*b = 0`	`a-2*b == 0`
d.	`NOT (a = 2*b)`	`!(a == 2*b)`

2. Pentru subprogramul **f** cu definiția următoare, indicați ce se va afișa în urma apelului **f(5427)**.

Limbajul Pascal

```
function f(x:longint):longint;
begin
 if x=0 then f:=0
  else
   if x mod 2=1 then
    f:=f(x div 10)*10+x mod
10-1
    else
    f:=f(x div 10)*10+x mod
10+1
end end
```

Limbajul C/C++

```
int f(int x)
{if(x==0)return 0;
 if(x%2==1)
   return
    f(x/10)*10+x%10-1;
   else
    return
    f(x/10)*10+x%10+1;}
```

a. 4563	b. 4536	c.6345	d. 6354

3. La generarea numerelor formate cu **n** cifre cu elementele mulţimii **{0,4,7}** se utilizează un algoritm backtracking care, pentru **n=2**, generează, în ordine, numerele **40, 44, 47, 70, 74, 77**. Dacă **n=4** şi se utilizează acelaşi algoritm, care este numărul generat imediat după numărul **4007**?

a. 7000	b. 4040	c. 7040	d. 7004

4. Se consideră un arbore cu 10 noduri, numerotate de la 1 la 10, şi cu vectorul "de taţi" următor: (0,1,1,1,3,3,3,4,7,7). Fii nodului 3 sunt:

a. 5,6,7	b. 6,7	c. 5,6,8	d. 5,7

5. Se consideră un graf neorientat cu 54 noduri şi 44 muchii. Care este numărul maxim de noduri izolate pe care le poate avea graful?

a. 44	b. 46	c. 45	d. 50

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **a%b** restul împărţirii numărului întreg **a** la numărul întreg nenul **b** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte n (număr natural)
max<-1
┌cât timp n>0 execută
│┌dacă max<n%10 atunci
││   max<-n%10
│└■
│ n<-[n/10]
└■
scrie max
```

a) Ce valoare va fi afişată dacă se citeşte pentru **n=1454**? **(6p.)**

b) Scrieţi câte numere naturale formate din exact **3** cifre pot fi citite pentru variabila n astfel încat în urma executării algoritmului pentru fiecare dintre valorile citite se va afişa valoarea **4**. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **cât timp...execută** cu o structură repetitivă de alt tip. **(6p.)**

d) Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. Se consideră declarările următoare în care variabila **x** memorează informaţiile referitoare la nume, prenume si la cele **2** note ale unui elev. Care dintre instrucţiuni calculează în variabila reală **med**, media aritmetică a notelor elevului ale cărui informaţii sunt memorate in variabila **x**?

Limbajul Pascal	Limbajul C/C++
`type elev = record` `nume,prenume:char` `nota1,nota2:byte` `end;` `var x: elev;`	`struct elev{` `char nume[10],prenume[20];` `int nota1,nota2 }x;`

Scrieţi o expresie în limbajul **Pascal | C/C++** care să calculeze în variabila reală *med*, media aritmetică a notelor elevului ale cărui informaţii sunt memorate in variabila x. **(6p.)**

3. În secvenţa următoare, variabilele **i** şi **j** sunt de tip **integer|int**, iar variabila **A** memorează o matrice cu **4** linii şi **4** coloane (numerotate de la **1** la **4**) cu elemente de tip **integer|int**. Care este suma elementelor de pe diagonala principală a matricei **A** la finalul executării secvenţei? **(6p.)**

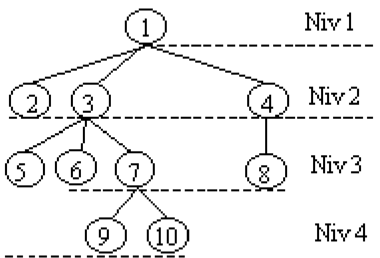
	Limbajul Pascal	Limbajul C/C++
	`for i:= 1 to 4 do` `for j:= 1 to 4 do` `if (i+j)mod 2=0 then` `A[i,j]:=(i+j) div 4` `else` `A[i,j]:=2*i+j;`	`for(i=1;i<=4;i++)` `for(j=1;j<=4;j++)` `if((i+j)%2==0)` `A[i][j]=(i+j)/4;` `else` `A[i][j]=2*i+j;`

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Scrieţi definiţia completă a unui subprogram sub, cu trei parametri: -v, un tablou unidimensional cu cel mult 100 de elemente numere întregi cu cel mult 5 cifre fiecare; - a si b, numere naturale intre 1 si 100. Subprogramul caută primul element divizibil cu 3 în secvenţa v[a], v[a+1], ..., v[b], şi returnează poziția acestuia, dacă există un astfel de element sau valoarea **-1** în caz contrar. **(10p.)**
2.	Scrieţi un program care citeşte de la tastatură un număr natural **n** (**n**<**100**) si apoi **n** cuvinte, câte unul pe linie. Cuvintele sunt formate fiecare din minimum două litere şi maximum **20**, doar litere mici ale alfabetului englez. Programul va afişa pe ecran numărul de cuvinte care rimează cu primul cuvânt citit. Spunem că două cuvinte rimează dacă au ultimele două caractere identice. ***Exemplu***. Pentru cuvintele: care masa mare casa test lucrare se va afişa: **2** deoarece cuvantul care rimează cu cuvintele mare si lucrare. **(10p.)**
3.	Fişierul text bac.in conţine pe prima linie două numere naturale **x** şi **y**, **x**<**y** (formate din cel mult **9** cifre fiecare), separate printr-un spaţiu. **a**) Scrieţi un program care citeşte numerele **x** şi **y** şi apoi, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină şi afişează pe ecran câte dintre numerele din intervalul **[x,y]** au proprietatea că atât ele cât şi răsturnatele lor sunt pătrate perfecte. **(8p.)** ***Exemplu*****:** dacă fişierul *bac.in* are conţinutul: **5 1000** atunci, pe ecran se va afişa numărul **9** reprezentând numărul de valori din intervalul **[5,1000]** care au proprietatea cerută şi anume: **9, 121, 144, 169, 441, 484, 676, 900, 961**. **b)** Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

VARIANTA 9 - *Rezolvare*

Subiectul 1

<table>
<tr><td>1.</td><td>Se accesează valorile memorate în variabilele a și b folosind operatorii mod|%,-, *.
Pentru precizarea răspunsului d) se acordă 4p.</td></tr>
<tr><td>2.</td><td>Se urmăresc apelurile recursive și se fac calculele:
f(5427)=f(542)*10+7-1
f(542)=f(54)*10+2+1
f(54)= f(5)*10+4+1
f(5)= f(0)*10+5-1
f(0)=0
⇨ f(5)=0*10+5-1=4
⇨ f(54)= f(5)*10+4+1=45
⇨ f(542)= f(54)*10+2+1=453
⇨ f(5427)= f(542)*10+7-1=4536
Pentru precizarea răspunsului b) se acordă 4p.</td></tr>
<tr><td>3.</td><td>Se generează în aceeași manieră ca în exemplul din enunț.
Pentru răspunsul b) se acordă 4p.</td></tr>
<tr><td>4.</td><td>Din vectorul de referințe ascendente deducem că rădăcina arborelui este nodul 1 (tata[1]=0). Observăm că nodurile 5,6,7 sunt fii nodului 3.

Pentru precizarea răspunsului a) se acordă 4p.</td></tr>
<tr><td>5.</td><td>Putem folosi 10 noduri pentru a face un graf complet cu 45 de muchii sau cu 9 noduri pentru a face un graf cu 36 de muchii. Cum noi avem 44 de muchii, le putem folosi cu 10 noduri. Deci ne rămân 44 de noduri cu grad 0.
Pentru precizarea răspunsului a) se acordă 4p.</td></tr>
</table>

Subiectul 2

1.

a) Se compară succesiv cifrele lui **n** cu valoarea variabilei **max** inițializată la inceput cu -1. Pentru a avea acces la toate cifrele lui **n**, se incepe compararea cu cifra unitaților si apoi se elimină acestă cifră repetând procedeul pănă când **n** devine **0**. Variabila **max** va memora cea mai mare cifră a numărului **n**.

Pentru precizarea valorii **5** se acordă **6p**.

b) Pentru variabila **n** poate fi citit orice număr natural de trei cifre care are prima cifră diferită de **0**. Sunt 4 valori posibile pentru prima cifră, **5** valori posibile pentru a doua cifră și **5** valori posibile pentru a treia cifră. În total sunt

1x5x5+4x1x5+4x5x1=65

numere care pot fi citite pentru variabila **n**.

Pentru precizarea valorii **65** se acordă **4p**.

c) Înlocuirea structurii `cât timp n>0 execută...` presupune utilizarea unei structuri repetitive **repetă - cât timp** sau **repetă - până când**:

```
 citeşte n (număr
natural)
 max<-1
 ┌dacă n>0 atunci
 │┌repetă
 ││┌dacă max<n%10 atunci
 │││   max<-n%10
 ││└■
 ││ n<-[n/10]
 │└cat timp n>0
 └■
 scrie max
```

```
citeşte n (număr
natural)
 max<-1
 ┌dacă n>0 atunci
 │┌repetă
 ││┌dacă max<n%10 atunci
 │││   max<-n%10
 ││└■
 ││ n<-[n/10]
 │└pana cand n=0
 └■
 scrie max
```

Pentru alegerea unei structuri repetitive se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 2p, iar pentru scrierea integrală a restului algoritmului se mai acordă 2p. – în total **6p**.

d) Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului şi de decizie se acordă câte 1p., pentru instrucţiunea repetitivă se acordă 3p., iar pentru celelalte atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.

2.

Un răspuns corect:

Pascal:**med:=(x.nota1+x.nota2)/2;**

C/C++: **med =(x.nota1+x.nota2)/2.0;**

Pentru orice răspuns corect se acordă **6p**.

3.	La finalul executării secvenței date conținutul matricei **A** este: **0 4 1 7** **5 1 7 1** **1 8 1 10** **9 1 11 2** Se observă că suma elementelor de pe diagonala principală este **4 (=0+1+1+2**) Pentru scrierea numărului **4** se acordă **6p.**

Subiectul 3

1.

Limbajul Pascal

```
function sub(n,a,b:byte;v:vector):integer;
var i:byte;rezultat:integer;
begin
    i:=a;
    while (i<=b) and (v[i] mod 3 <>0) do
      if i=b+1 then rezultat:= -1
        else rezultat:= i;
end;
```

Limbajul C/C++

```
int sub(int v[100], int n, int a, int b)
{int i;
i=a;
while(i<=b && a[i]%3!=0) i++;
if(i==b+1) return -1;
else return i;
}
```

Pentru antet corect se acordă 2p. Pentru condiţia divizibilitaţii din tabloul a se acordă 2p. Pentru folosirea corectă a structurii **while** se dau 2p, iar pentru determinarea poziției se acordă 2p. Pentru corectitudinea globală a subprogramului 2p. – în total **10p**

2.

Pentru rezolvarea acestei probleme ca soluție se propune scrierea unei funcții care primește două șiruri ca parametri și testează dacă acestea rimează. Apoi ar trebui să numărăm pentru primul cuvânt cu ce cuvinte din șir rimează (mai exact cu câte cuvinte care sunt după el în șir rimează).

Se pot utiliza funcţiile predefinite pentru şiruri de caractere specifice limbajului.

Pentru declararea corectă a variabilelor se acordă 2p. Pentru citirea șirului se acordă 2p. Pentru numărarea aparițiilor sufixelor de două litere identice cu primul sufix se acordă 2p. Pentru afișarea numărului de cuvinte cu proprietatea cerută se acordă 2p.

Pentru corectitudinea globală a programului se acordă 2p – în total **10p**.

Limbajul Pascal

```
var s,t:string;
l1,l2,k,n,i:byte;
```

<table>
<tr><td></td><td colspan="2">

```
 begin
  readln(n);
  readln(s);
  l1:=length(s);

  for i:=2 to n do begin
    readln(t);
    l2:=length(t);
    if (s[l1-1]=t[l2-1]) and (s[l1]=t[l2]) then inc(k);
  end;
  writeln(k);
 end.
```

Limbajul C/C++

```
#include <iostream>
#include <cstring>
using namespace std;
int main()
{int n;
 cin >> n;
 char cuv1[21], cuv2[21];
 cin  >> cuv1;
 int s = strlen(cuv1);
 int nr = 0;
 for(int i = 1; i < n; i++)
  { cin >> cuv2;
    int s2 = strlen(cuv2);
    if(cuv2[s2-2]==cuv1[s-2] && cuv2[s2-1]==cuv1[s-1])
       nr++;
   }
  cout << nr;
  return 0;
}
```

</td></tr>
<tr><td>3.</td><td>b)</td><td>

Algoritmul de rezolvare propus parcurge toate numerele de la radical din **x** până la radical din **y**, apoi construiește răsturnatul fiecărui pătrat perfect și verifică dacă și răsturnatul este pătrat perfect.

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul de numere. Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple, fără folosirea unor structuri suplimentare de date.

Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. – în total 2p.

</td></tr>
<tr><td></td><td>a)</td><td>

Limbajul Pascal

```
var f:text;
    a,b,x,y,r,i,k:longint;
begin
  assign(f,'bac.in');reset(f);
  read(f,x,y);
  k:=0;
```

</td></tr>
</table>

```
   a:=x;
   x:=trunc(sqrt(a));
   if x*x<a then inc(x);
   y:=trunc(sqrt(y));
   for i:=x to y do begin
          a:=i*i; r:=0;
          while a>0 do begin
               r:=r*10+a mod 10;
               a:=a div 10
          end;
       b:=trunc(sqrt(r));
       if (x<=b) and (b<=y) and (b*b=r) then k:=k+1;
   end;
   writeln(k);
   close(f);
 end.
```

Limbajul C/C++

```
#include <iostream>
#include<fstream>
#include <cmath>
using namespace std;
ifstream f("bac.in");
int main()
{   int a,b, x,y,i,r,k=0;
    f>>x>>y;
    a=x;
    x=(int)sqrt(a);
    if(x*x<a)x++;
    y=(int)sqrt(y);
    for(int i=x;i<=y;i++)
    {   a=i*i; r=0;
        while(a>0)
        {   r=r*10+a%10;
            a=a/10;
        }
        b=(int)sqrt(r);
        if(x<=b && b<=y && b*b==r) k++;
    }
    cout<<k;
    f.close();
    return 0;
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea valorii cerute, 1p. pentru afișarea ei, 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare și 1p. pentru utilizarea eficientă a memoriei. – în total 8p.

VARIANTA 10

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 punc

1. Indicaţi expresia care are valoarea **true/1** dacă şi numai dacă numărul real memorat în variabila **x** se află în intervalul `[-5,5]`.

	Limbajul Pascal	Limbajul C/C++
a.	`x*y-8<> 0`	`x*y-8!= 0`
b.	`x*(y-5) > 0`	`x*(y-5) > 0`
c.	`x*(y-5)= 0`	`x*(y-5)== 0`
d.	`x*y-9 < 0`	`x*y-9 < 0`

2. Pentru subprogramul **f** cu definiția următoare, indicați ce se va afișa în urma apelului **f(6359).**

Limbajul Pascal

```
function f(n:integer):integer;
var c:integer;
begin
if n=0 then f:=0
else begin
  c:= f(n div 10);
  if n mod 10>c then
       f:=n mod 10
   else f:=c
end end
```

Limbajul C/C++

```
int f(int n){
if (n==0) return 0;
else
{int c=f(n/10);
 if (n%10>c)
  return n%10;
 else return c;
}}
```

a. 3	b. 9	c.0	d. 5

3. Un algoritm de tip backtracking generează în ordine crescătoare toate numerele de câte cinci cifre din mulțimea {2,3} și cu proprietatatea că nu există mai mult de două cifre 2 alăturate. Primele 6 solutii generate sunt: 22322, 22323, 22332, 22333, 23223, 23232. Care este cea de-a 8 a solutie generată?

a. 23233	b. 32233	c. 23322	d. 32322

4. Se consideră un arbore cu 8 noduri, numerotate de la 1 la 8, şi cu vectorul "de taţi" următor: (2 0 2 5 2 5 3 7). Frunzele arborelui sunt:

a. 4, 6, 8	b. 1,4,6	c. 1, 4, 6, 8	d. 1, 4, 8

5.	Se consideră un graf neorientat dat prin matricea de adiacenţă următoare: 0 1 1 0 0 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 0 0 1 1 0 Care este numărul maxim de muchii care pot fi eliminate din graf astfel încât graful parţial rezultat să fie conex ?			
	a. 1	**b.** 2	**c.** 3	**d.** 4

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **a%b** restul împărţirii numărului întreg **a** la numărul întreg nenul **b** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte n (număr natural)
r<-0;
┌cât timp n>0 execută
│ r<-r*10 + n%10;
│ n<-[n/10]
└■
┌cât timp r>0 execută
│┌dacă r%2=0 atunci
││ n<-n*10 +r%10
│└■
│ r<-[r/100]
└■
scrie n
```

a)	Ce valoare va fi afişată dacă se citeşte pentru `n` valoarea `23758`? **(6p.)**
b)	Scrieţi un număr natural format din exact 5 cifre care poate fi citit pentru variabila n astfel încat in urma executării algoritmului să se afiseze valoarea 4. **(6p.)**
c)	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **`cât timp...execută`** cu o structură repetitivă cu test final. **(6p.)**
d)	Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2.	Două puncte A şi B din planul xOy sunt date prin coordonatele lor carteziene a,b, respectiv x, y. Scrieți o expresie care are valoarea 1 dacă și numai dacă punctele A și B se află pe a II a bisectoare a sistemului de coordonate. **(6p.)**
3.	Considerăm următoarea secvență de program. Scrieţi ce se va afişa pe ecran în urma executării secvenței de instrucțiuni de mai sus, pentru şirul **calcul**. **(6p.)**

	Limbajul Pascal	Limbajul C/C++
	`var s:string;` `     n,i:integer;` `begin` `.................` ` readln(s);` ` n:=length(s);` ` for i:= 1 to n do` `    writeln(copy(s,1,i);` ` readln;` `end.`	`char s[30];` `int i,j;` `..........................................` `cin.getline(s,30);` `i=0;` `while(s[i]!=0)` `{ for(j=0;j<=i;j++)` `     cout<<s[j];` `  cout<<endl;` `  i++;` `}`

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Scrieţi definiţia completă a unui subprogram **suma**, cu doi parametri: - **v**, prin care primeşte un tablou unidimensional cu n elemente, numere intregi situate în intervalul [10,25000]; - **n**, prin care primeşte un număr natural (1≤n≤100);. Subprogramul returnează suma numerelor din tabloul v care au ultimele 2 cifre pare. **Exemplu: dacă n=5 şi v=(124, 122, 423, 555, 1046) funcţia va returna 1292 (=124+122+1046).** **(10p.)**
2.	Scrieţi programul care citeşte de la tastatură o valoare naturală n (2<n<30) construieşte în memorie şi afişează pe ecran o matrice cu proprietatea că: -fiecare element din matrice aflat pe o linie impară este egal cu produsul dintre indicele liniei şi indicele coloanei pe care se află -fiecare element aflat pe o linie pară este egal cu suma dintre pătratul indicelui coloanei pe care se află şi elementul de deasupra lui. Elementele matricei vor fi afişate pe linii, cu spaţii între elementele de pe fiecare linie. *Exemplu*: pentru n=5 se va afişa matricea următoare:

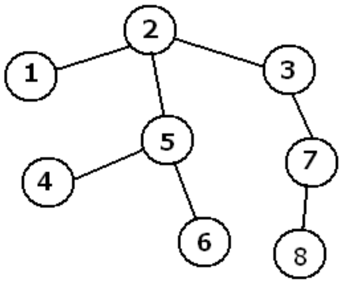
	1 2 3 4 5 2 6 12 20 30 3 6 9 12 15 4 10 18 28 40 5 10 15 20 25 **(10p.)**
3.	Fişierul text *bac.in* conţine între 2 şi cel mult 10000 de numere naturale distincte, dintre care cel puţin două au ultima cifră divizibilă cu 3. Numerele sunt separate prin câte un spaţiu şi fiecare dintre ele are cel mult 9 cifre. **a**) Scrieţi un program care determină cele mai mari două numere din fişier care au ultima cifră divizibilă cu 3, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat. Cele două numere vor fi afişate pe o linie a ecranului, în ordine crescătoare, separate printr-un spaţiu. **(8p.)** **Exemplu:** dacă fişierul *bac.in* are conţine numerele: 5126 8 6 12 3 23 443 atunci, pe ecran se vor afişa numerele: 443 5126 **b)** Descrieţi succint, în limbaj natural, algoritmul utilizat, justificând eficienţa acestuia. **(2p.)**

⚠ **Observaţii:**

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 10 - *Rezolvare*

Subiectul 1

1.	Se accesează valorile memorate în variabilele **x** și **y**. Pentru precizarea răspunsului **d)** se acordă **4p.**
2.	Se urmăresc apelurile recursive și se fac calculele. Se compară două câte două si se va afișa valoarea maximă dintre cifre. Pentru răspunsul **b)** se acordă **4p**.
3.	Se generează în aceeasi manieră ca in exemplul din enunt. Pentru răspunsul **c)** se acordă **4p**.
4.	Frunza = într-un arbore cu radacina nodul x este frunză daca nu are nici un descendent direct. Observăm că nodurile **1,4,6,8** sunt frunze.  Pentru precizarea răspunsului **c)** se acordă **4p.**
5.	Observăm ca se pot elimina maxim 3 muchii. Putem elimina muchiile (1,3), (2,4), (5,3) si de aici rezulta un graf conex. Pentru precizarea răspunsului c**)** se acordă **4p.**

Subiectul 2

1.	**a)**	Algoritmul calculează numărul obținut din cifrele lui pare aflate pe poziții impare, numărarea pozițiilor cifrelor începand cu cifra cea mai semnificativă. Pentru precizarea valorii **28** se acordă **6p**.
	b)	Pentru variabila **n** poate fi citit orice număr natural de cinci cifre care are pe una din pozițiile impare, o singură cifră egală cu valoarea **4**. Pe celelalte 2 poziții pot fi oricare alte 2 cifre impare. Un exemplu de număr: 13475. Pentru precizarea valorii corecte se acordă **6p**.
	c)	Înlocuirea structurii **cât timp n>0 execută ...** presupune utilizarea unei structuri repetitive **repetă – cât timp** sau, **repetă - până când** :

```
citeşte n(număr natural)
r<-0;
┌daca n>0 atunci
│┌repetă
││ r<-r*10 + n%10;
││ n<-[n/10]
│└ cât timp n>0
│┌repetă
││┌dacă r%2=0 atunci
│││ n<-n*10 +r%10
││└■
││ r<-[r/100]
│└cât timp r>0
└■
scrie n
```

```
citeşte n(număr
natural)
r<-0;
┌daca n>0 atunci
│┌repetă
││ r<-r*10 + n%10;
││ n<-[n/10]
│└ până când n=0
│┌repetă
││┌dacă r%2=0 atunci
│││ n<-n*10 +r%10
││└■
││ r<-[r/100]
│└până când r=0
└■
scrie n
```

Pentru alegerea unei structuri repetitive se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 2p, iar pentru scrierea integrală a restului algoritmului se mai acordă 2p. – în total **6p**.

d) Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc.

Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și de decizie se acordă câte 1p., pentru instrucţiunile repetitive se acordă 3p., iar pentru celelalte atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.

2. **a==-b && x==-y**

Pentru precizarea răspunsului corect se acordă **6p.**

3. În urma executării instrucțiunilor din secvența dată se vor afisa prefixele cuvântului :

c
ca
cal
calc
calcu
calcul

Pentru precizarea răspunsului corect se acordă **6p**.

Subiectul 3

1.	**Limbajul Pascal** ```Type vector=array[1..100]of integer;``` ```function suma(n:byte;v:vector):integer;``` ```var ult,pult,i:byte;``` ``` s:integer;``` ```begin``` ``` s:=0;``` ``` for i:=1 to n do``` ``` begin``` ``` ult:=v[i] mod 10;``` ``` pult:=v[i] div 10 mod 10;``` ``` if (ult mod 2=0) and (pult mod 2=0) then``` ``` s:=s+v[i];``` ``` end;``` ``` suma:=s;``` ```end;``` **Limbajul C/C++** ```int suma(int n,int v[])``` ```{ int i,s=0,ult,pult;``` ``` for(i=1;i<=n;i++)``` ``` { ult=v[i]%10;``` ``` pult=v[i]/10%10;``` ``` if (pult%2==0 && ult%2==0) s+=v[i];``` ``` }``` ``` return s;``` ```}``` **Pentru** antet corect se acordă 2p. Pentru condiţia divizibilitaţii din tabloul v se acordă 2p. Pentru folosirea corectă a structurii for se dau 2p, iar pentru determinarea ultimelor două cifre se acordă 2p. Pentru corectitudinea globală a subprogramului 2p. În total **10p.**
2.	Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui n se acordă 1p., pentru completarea corectă a fiecărui element din matrice aflat pe o linie impară 2p., pentru completarea corectă a fiecărui element din matrice aflat pe o linie pară 2p.,iar pentru afișarea matricei conform cerinței se acordă 2p. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă 2p. - în total 10p. Pot fi concepute mai multe metode de obținere a matricei cerute. Metoda clasică ce completează coloanele matricei, cu liniile, respectiv coloanele, numerotate de la **1** la **n**, respectiv de la **1** la **n**, în maniera descrisă, este: **Limbajul Pascal** ```var a: array[1..30,1..30] of integer;``` ```n,i,j:byte;```

```
begin
 for i:=1 to n do
  for j:=1 to n do
   if i mod 2<>0 then
     a[i,j]:=i*j
    else
     a[i,j]:=j*j+a[i-1,j];
 for i:=1 to n do
  begin
    writeln;
    for j:=1 to n do
     write(a[i,j],' ')
  end
end.
```

Limbajul C/C++

```
#include<iostream>
using namespace std;

int main()
{ int i,j,n,a[30][30];
  cin>>n;

  for(i=1;i<=n;i++)
  {for(j=1;j<=n;j++)
  if(i%2!=0)
      a[i][j]=i*j;
      else
      a[i][j]=j*j + a[i-1][j];
  for(i=1;i<=n;i++)
   {for(j=1;j<=n;j++)
       cout<<a[i][j]<<" ";
    cout<<endl;
   }

  return 0;
}
```

3. b) Se citesc pe rând valorile din fişier, verificăm dacă se îndeplineşte proprietatea cerută şi în caz afirmativ, îl comparăm cu nr2 respectiv cu nr1, înlocuindu-le când e cazul.

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată şirul de numere (complexitate O(n)). Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple, fără folosirea unor structuri suplimentare de date (complexitate O(1)).

Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. – în total 2p.

a)

Limbajul Pascal

```
var x,nr1,nr2:longint;
f:text;
```

```
begin
   assign (f,'bac.in');reset(f);
   nr1:=0;
   nr2:=0;
   while not eof(f) do
    begin read(f,x);
     if x mod 10 mod 3=0 then
      if nr2<x then
       begin
          nr1:=nr2;
          nr2:=x;
       end
               else
         if nr1<x then
               nr1:=x;
          end;
   write(nr1,' ',nr2);
  close(f);
end.
```

Limbajul C/C++

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{ int x,nr1,nr2;
  ifstream f("bac.in");
  nr1=nr2=0;
  while(f>>x)
  { if(x%10%3==0)
     if(nr2<x)
     { nr1=nr2;
       nr2=x;
     }
     else
       if(nr1<x)
          nr1=x;
  }
  cout<<nr1<<" "<<nr2;
  return 0;
}
```

Se acordă **1p**. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă **1p**. pentru citirea tuturor numerelor din fişier, **1p**. pentru un algoritm principial corect, încă **1p**. pentru determinarea valorilor cerute, **1p**. pentru afișarea lor, **1p**. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă **1p**. pentru alegerea unui algoritm eficient ca timp de executare și **1p**. pentru utilizarea eficentă a memoriei. În total **8p.**

VARIANTA 11

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Indicaţi expresia care are valoarea **true/1** dacă şi numai dacă valorile variabilelor întregi pozitive **a** şi **b** au aceeaşi paritate (sunt ambele pare sau ambele impare).

	Limbajul Pascal	Limbajul C/C++
a.	`a div 2 = b div 2`	`a/2 == b/2`
b.	`a mod 2 - b mod 2 = 0`	`a%2 - b%2 == 0`
c.	`(a mod 2)*(b mod 2) = 0`	`(a%2)*(b%2) == 0`
d.	`(a - b) mod 2 = 1`	`(a - b) % 2 == 1`

2. Pentru subprogramul **f** cu definiția următoare, indicați ce se va afișa în urma apelului **f(3)**?

Limbajul Pascal

```
procedure f(n: integer);
var i: integer;
begin
  dec(n);
  for i := 1 to n do
  begin
    f(i);
    write(i)
  end;
  write(n)
end;
```

Limbajul C/C++

```
void f(int n)
{
  int i;
  n--;
  for (i = 1; i <= n; i++)
  {
    f(i);
    printf("%d", i);| cout << i;
  }
  printf("%d", n);| cout << n;
}
```

a. 0101122	b. 010112	c.1021012	d. 121

3. Utilizând metoda backtracking se generează toate şirurile corecte de paranteze rotunde. Astfel, pentru **n=2**, se obţine şirul **()**, iar pentru **n=4**, se obţin, în ordine, şirurile: **(()), ()()**. Folosind aceeaşi metodă, se generează şirurile corecte formate din câte **6** paranteze.

Care va fi al patrulea şir generat?

a. ()(())	b. (())()	c. ()()()	d. (()())

4. Se consideră un graf neorientat **G** cu **10** vârfuri. Care este numărul minim de muchii pe care trebuie să le aibă **G** pentru a putea afirma cu siguranţă că **G** este conex?

a. 1	b. 9	c. 37	d. 45

5.	Se consideră un arbore cu **10** noduri, având rădăcina pe nivelul **0**. Ştiind că fiecare nod are cel mult **4** fii, care este numărul maxim de noduri care se pot afla pe nivelul **2**?			
	a. 5	b. 7	c. 8	d. 9

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte a, b (numere naturale nenule)
p<-0
┌pentru i<-a,b execută
│ j<-1
│┌cât timp j*j<i execută
││ j<-j+1
│└■
│┌dacă j*j=i atunci
││ p<-i
│└■
└■
scrie p
```

a) Ce se va afişa dacă se citesc valorile **14** şi **20**? **(6p.)**

b) Pentru **a=30** scrieţi toate valorile care pot fi introduse pentru variabila **b** astfel încât să se afişeze **0**. **(6p.)**

c) Scrieţi în pseudocod un algoritm echivalent cu cel dat, în care să se folosească o singură structură repetitivă. **(6p.)**

d) Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. Se consideră tipul de date **interval**, definit mai jos, capabil să memoreze extremităţile **a** şi respectiv **b** ale unui interval închis și variabilele **i** de tip **interval** și **x** de tip **real** (Pascal), respectiv **float** (C/C++) declarate mai jos.

Limbajul Pascal	Limbajul C/C++
`type interval = record` `       a,b:real` `end;` `var i: interval;` `    x: real;`	`typedef struct` `{` `   float a, b;` `} interval;` `interval i;` `float x;`

Scrieţi o secvenţă de instrucţiuni în urma executării căreia să se afişeze pe ecran mesajul **"DA"**, în cazul în care numărul real reţinut de variabila **x**, aparţine intervalului **i** şi respectiv mesajul **"NU"**, în caz contrar. **(6p.)**

3.	Completaţi punctele de suspensie din secvenţa de instrucţiuni de mai jos astfel încât în urma executării acesteia să se afişeze suma celor mai mici valori de pe fiecare coloană a matricei **a**, care are **m** linii şi **n** coloane. Atât liniile, cât şi coloanele matricei sunt numerotate începând cu **0**. **(6p.)**

Limbajul Pascal

```
s := 0;
for j := 0 to n - 1 do
  begin
    min := ...;
    for i := 1 to m - 1 do
      if a[..., ...] < min
        then min := a[..., ...];
    s := ...
  end;
writeln(s);
```

Limbajul C/C++

```
s = 0;
for (j = 0; j < n; j++) {
  min = ...;
  for (i = 1; i < m; i++)
    if (a[...][...] < min)
      min = a[...][...];
  s = ...;
}
printf("%d", s); | cout << s;
```

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. a) Scrieţi definiţia completă a subprogramului **radical**, cu un parametru, **n**, prin intermediul căruia primeşte un număr natural format din cel mult **9** cifre și returnează cel mai mare număr natural al cărui pătrat este mai mic sau egal cu **n**. De exemplu, apelul **radical(71)** va returna **8**, iar apelul **radical(100)** va returna **10**. **(4p.)**
b) Scrieți un program Pascal/C/C++ care, folosind apeluri utile ale subprogramului **radical** descris la punctul a), determină și afișează pe ecran cel mai mic număr mai mare sau egal cu o valoare **x** citită de la tastatură, care are proprietatea de a fi pătrat perfect.
Exemplu: dacă de la tastatură se citește **x=19**, atunci programul va afișa pe ecran **25**, iar dacă se citește **x=64**, se va afișa **64**. **(6p.)**

2. Scrieți un program care citeşte de la tastatură un text format din cel mult **100** de caractere, litere mici și mari, spații, caractere punct și caractere virgulă. Programul va modifica în memorie și va afișa textul obținut prin transformarea în litere mari (majuscule) a tuturor literelor mici care fac parte din cuvinte care încep și se termină cu aceeași literă. În verificarea egalității dintre prima și ultima literă a cuvântului nu se va face distincție între literele mici și mari. De exemplu, cuvântul **„Ieri”** va fi transformat în **„IERI”**. Cuvintele pot fi delimitate prin punct, virgulă, unul sau mai multe spații.
De exemplu, dacă se citeşte de la tastatură textul
Examenul este usor. Ieri am   invatat suficient.
se va construi în memorie şi se va afişa pe ecran:
Examenul ESTE usor. IERI am   invatat suficient. **(10p.)**

3. Fişierul text **bac.in** conţine pe prima linie două numere naturale **n** şi **x** **(2≤n≤100000,1≤x≤2000000000)**, separate printr-un spațiu, iar pe următoarea linie **n** numere naturale de cel mult **9** cifre fiecare, separate prin câte un spațiu, **ordonate strict crescător**.

a) Scrieţi un program care, utilizând un algoritm eficient din punctul de vedere al timpului de executare determină şi afişează pe ecran câte perechi formate cu numere aflate pe cea de-a doua linie a fișierului au suma celor **2** elemente care le compun mai mică sau egală cu **x**. **(8p.)**

Exemplu: dacă fişierul **bac.in** are conţinutul:

9 10

1 2 3 4 6 7 8 9 11

atunci se afişează **16** deoarece perechile care îndeplinesc condiția cerută sunt: **(1,2),(1,3),(1,4),(1,6),(1,7),(1,8),(1,9),(2,3),(2,4),(2,6),(2,7),(2,8),(3,4),(3,6),(3,7)** şi **(4,6)**

b) Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

Observaţii:

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 11 - *Rezolvare*

Subiectul 1

1.	Numerele cu aceeași paritate dau același rest la împărțirea cu `2`. Prima variantă este incorectă pentru că valorile **`a div 2`**, **`b div 2`** și respectiv **`a/2`**, **`b/2`** corespund câturilor la împărțirea cu `2`. Un contraexemplu este dat de **`a=4`**, **`b=5`**, pentru care expresia e adevărată deși numerele au parități diferite. Varianta **b** este cea corectă. Expresia de la varianta **c** are valoarea **`true/1`** dacă și numai dacă cel puțin unul dintre numerele reținute de **`a`** și **`b`** este par. Contraexemple: pentru **`a=1`**, **`b=2`** are valoarea **`true/1`**, iar pentru **`a=3`**, **`b=5`** are valoarea **`false/0`.** Expresia de la varianta **d** este totdeauna falsă dacă **`a`** și **`b`** au aceeași paritate. Pentru precizarea răspunsului **b)** se acordă **4p**.
2.	Înaintea oricărei afișări, **`f(3)`** apelează **`f(1)`**. În apelul **`f(1)`** nu se va intra în bucla **`for`**, prin urmare nu se vor face alte apeluri recursive; la final se va afișa pe ecran `0`. La revenirea din apelul **`f(1)`**, în **`f(3)`** se va afișa **`i`**, adică **`1`**, apoi se va trece la următorul pas din **`for`** și se va apela **`f(2)`**. Bucla **`for`** din **`f(2)`** se va executa doar o dată. În cadrul acesteia se va apela **`f(1)`**, în cadrul căruia se va afișa `0`, apoi la revenire se va afișa **`1`**, iar la final se va mai afișa **`n`**, adică din nou **`1`**. Apoi se va reveni în apelul **`f(3)`**, care va afișa valoarea curentă a lui **`i`**, adică **`2`**, iar la final, după încheierea buclei, pe cea curentă a lui **`n`**, adică tot **`2`** și se va încheia. Prin urmare șirul afișat este `0101122`. Pentru răspunsul **a)** se acordă **4p**.
3.	Conform precizărilor din enunț, pe fiecare poziție în parte se încearcă mai întâi plasarea unei paranteze deschise, iar apoi, dacă acest lucru nu mai e posibil, a uneia închise. Astfel, pentru **`n=6`** se generează în ordine soluțiile: **`((()))`**, **`(()())`**, **`(())()`**, **`()(())`**, **`()()()`**. A patra soluție este **`()(())`**. Pentru răspunsul **a)** se acordă **4 p.**
4.	Numărul maxim de muchii pe care le poate avea un graf neorientat cu `10` vârfuri care **nu** este conex este **`36`**. La acest rezultat se ajunge păstrând un vârf izolat și luând toate muchiile posibile care nu îl au ca extremitate. Prin urmare, pentru a avea certitudinea că graful este conex trebuie să avem cel puțin **`1+36=37`** de muchii. Pentru răspunsul **c)** se acordă **4p.**
5.	Pentru fiecare grup de cel mult **`4`** noduri aflate pe nivelul **`2`** al arborelui este necesar să avem un tată al acestora pe nivelul **`1`**. În plus, pe nivelul **`0`** vom avea rădăcina. Putem construi un arbore cu **`7`** noduri pe nivelul **`2`**. Vectorul de tați ar putea fi **`(0,1,1,2,2,2,2,3,3,3)`**. Putem de asemenea demonstra că e imposibil să avem mai mult de **`7`** noduri pe ultimul nivel. Dacă am lua, de exemplu, **`8`** noduri pe nivelul **`2`**, ar mai fi necesare cel puțin **`2`** noduri pe nivelul **`1`** și un nod pe nivelul **`0`** (rădăcina), deci **`11`** noduri în total. Pentru răspunsul **b)** se acordă **4p.**

Subiectul 2

1.

a) Algoritmul are ca efect afișarea celui mai mare pătrat perfect din intervalul `[a,b]`. Dacă intervalul nu conține niciun astfel de număr, atunci se va afișa `0`. În intervalul `[14,20]` există un singur pătrat perfect, `16`. Aceasta va fi valoarea afișată.

Pentru precizarea valorii `16` se acordă **6p**.

b) Pentru **`a=30`**, valorile posibile pentru **`b`** sunt **`31, 32, 33, 34, 35`**, deoarece **`36`** este pătrat perfect și dacă **`b`** ar fi mai mare strict decât **`35`**, în intervalul dat ar exista cel puțin un pătrat perfect, prin urmare nu s-ar mai afișa `0.` Rezultatul poate fi găsit printr-o abordare exhaustivă: se încearcă pe rând pentru **`b`** primele valori mai mari strict ca **`a`**. Atunci când se ajunge la **`36`**, se constată că pentru orice valoare mai mare sau egală a lui **`b`**, nu se va mai putea afișa `0`.

Rezultatul poate fi calculat imediat dacă se înțelege prelucrarea din interiorul buclei **`pentru`**: la sfârșitul lui **`cât timp`**, **`j`** va fi cel mai mic număr natural al cărui pătrat e mai mare sau egal cu **`i`**. Condiția **`j*j=i`** corespunde de fapt verificării dacă `i` este pătrat perfect.

Pentru precizarea valorilor **`31, 32, 33, 34, 35`**, se acordă **6p**.

c) Un algoritm echivalent cu cel dat este următorul: calculăm **`r`**=cel mai mare număr care ridicat la pătrat nu depășește **`b`**. Dacă **`p=r*r`** e mai mare sau egal cu **`a`**, atunci **`p`** va fi rezultatul afișat, altfel scriem **`0`**.

```
citeşte a,b (numere naturale nenule, a<b)
 r←1
┌cât timp r*r≤b execută
│ r←r+1
└■
 r←r-1
 ┌dacă r*r≥a atunci
 │  p←r
 │altfel
 │  p←0
 └■
 scrie p
```

Pentru verificarea proprietății de a fi pătrat perfect a numărului determinat se acordă **1 p**., pentru verificarea dacă numărul găsit aparține intervalului **`[a,b]`** se acordă **1p.**, pentru tratarea cazului în care intervalul nu conține pătrate perfecte se acordă **1p**., iar pentru folosirea corectă a unei singure structuri repetitive se mai acordă **3p.** – în total **6p**.

d) Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, experiențe care

		au cimentat cunoștințele privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor etc. Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului şi de decizie se acordă câte **1p.**, pentru cele două instrucţiuni repetitive se acordă **3p.**, iar pentru cele **4** atribuiri se acordă **2p.** Pentru structura corectă a programului se mai acordă **1p.**, în total **10p**.

2. Pentru ca **`x`** să aparțină intervalului **`i`** este necesar ca **`x`** să respecte simultan condițiile **`x>=i.a`** și **`x<=i.b`**. Secvența de instrucțiuni cerută este:

Limbajul Pascal

```
if (x>=i.a) and (x<=i.b) then
  write('DA')
else
  write('NU')
```

Limbajul C/C++

```
if(x>=i.a && x<=i.b)
  printf("DA");
    | cout<<"DA";
else
  printf("NU");
    | cout<<"NU";
```

Pentru accesarea corectă a câmpurilor înregistrării se acordă **4** puncte (câte un punct pentru fiecare câmp).
Pentru verificarea condiției de apartenență la interval se acordă **1** punct.
Pentru afișarea mesajelor cerute se acordă **1** punct.
În total, pentru o secvență de instrucțiuni care respectă cerința se acordă toate cele **6 puncte**.

3. Pentru a completa secvența de instrucțiuni conform cerinței, trebuie observat că o variantă corectă de inițializare a lui **`min`** este de a-i atribui valoarea primului element al coloanei **`j`**, **`a[0,j]`** (respectiv **`a[0][j]`**). Elementele coloanei **`j`** sunt de forma **`a[i,j]`** (respectiv **`a[i][j]`**). După ce se calculează rezultatul specific fiecărei coloane, reținut în variabila **`min`**, se actualizează **`s`**, care crește cu valoarea lui **`min`**.

Limbajul Pascal

```
s:=0;
for j:=0 to n-1 do
  begin
    min := a[0,j];
    for i:=1 to m-1 do
      if a[i,j] < min
        then
          min:=a[i, j];
    s:=s+min
  end;
writeln(s);
```

Limbajul C/C++

```
s=0;
for (j=0; j<n; j++){
  min=a[0][j];
  for (i=1; i<m;i++)
    if (a[i][j]<min)
      min=a[i][j];
  s=s+a[i][j];
}
printf("%d", s);
      | cout << s;
```

Se acordă câte **2** puncte pentru fiecare completare corectă a punctelor de suspensie, înlocuirea cu **`a[i,j]`** (respectiv **`a[i][j]`**) fiind punctată o singură dată – în total **6 puncte**.

Subiectul 3

1. a) Pentru a calcula partea întreagă inferioară a lui radical din **n**, care este de fapt rezultatul pe care trebuie să îl returneze funcția **radical**, este suficientă o căutare secvențială a numărului cerut. Mai precis, vom găsi cel mai mic număr **r**, al cărui pătrat depășește **n** (cât timp **r*r** e prea mic, îl incrementăm pe **r**) și vom returna **r-1** (care este cu siguranță cel mai mare număr natural al cărui pătrat e mai mic sau egal cu **n**).

Problema admite o soluție mai eficientă, bazată pe căutarea binară, dar aceasta nu e recomandată în condițiile unui examen scris, implementarea fiind ceva mai dificilă.

Pentru antet corect se acordă **1 punct**, pentru determinarea rezultatului cerut se acordă **2 puncte**,iar pentru transmiterea (returnarea) acestuia **1 punct** – în total **4 puncte**.

Limbajul Pascal

```
function
radical(n:longint):
          longint;
var r:longint;
begin
  r:=0;
  while r*r<=n do
    inc(r);
  radical:=r-1;
end;
```

Limbajul C/C++

```
int radical(int n)
{
  int r=0;
  while(r*r<=n)
    r++;
  return r-1;
}
```

b) Programul principal cerut poate fi scris foarte ușor folosind un apel al funcției cerute la punctul **a)**. Se reține în variabila **rad** rezultatul lui **radical(x)** și apoi, pentru a trece de la partea întreagă inferioară la cea superioară, se incrementează **rad** dacă **x** nu este pătrat perfect. Rezultatul cerut este **rad*rad**.

Limbajul Pascal

```
var x,rad:longint;

begin
  read(x);
  rad:=radical(x);
  if rad*rad<x then
    inc(rad);
  write(rad*rad)
end.
```

Limbajul C/C++

```
#include <stdio.h>
int main()
{
  int x,rad;
  scanf("%d",&x);
  rad=radical(x);
  if(rad*rad<x)
    rad++;
  printf("%d",rad*rad;
  return 0;
}
```

Pentru declararea corectă a variabilelor se acordă **1** punct, pentru operațiile de intrare/ieșire se acordă **1** punct, pentru un apel corect și util al funcției de la punctul **a)** se acordă **2 puncte**, pentru determinarea rezultatului cerut (inclusiv tratarea cazului particular în care **x** este pătrat perfect) se acordă **1** punct, iar pentru corectitudinea globală a programului se acordă **1 punct** – în total **6 puncte**.

2. Există bineînțeles mai multe variante de rezolvare. Cea pe care o prezentăm mai jos are avantajul că poate fi adaptată ușor și altor cerințe asemănătoare legate de prelucrarea cuvintelor dintr-un text.

Limbajul Pascal:

```
var s:string;
    i,p,u:integer;

function litera(c:char):boolean;
var mica,mare:boolean;
begin
  mica:=((ord(c)>=ord('a'))and(ord(c)<=ord('z')));
  mare:=((ord(c)>=ord('A'))and(ord(c)<=ord('Z')));
  litera:=(mica or mare)
end;

function vocala(c:char):boolean;
var voc:string;
begin
  voc:='AEIOUaeiou';
  vocala:=(pos(c,voc)<>0)
end;

function literamica(c:char):boolean;
begin
  literamica:=((ord(c)>=ord('a'))
                and(ord(c)<=ord('z')))
end;

function majuscula(c:char):char;
var nr:integer;
begin
  nr:=ord(c)-ord('a');
  majuscula:=char(ord('A')+nr)
end;

begin
  readln(s);
  for i:=1 to length(s) do
    if litera(s[i]) then begin
        if (i=1)or(not litera(s[i-1]))
          then p:=i;
        if (i=length(s))or(not litera(s[i+1]))
          then begin
            u:=i;
            if (vocala(s[p]))and(vocala(s[u])) then
              while p<=u do begin
                  if literamica(s[p])
                    then s[p]:=majuscula(s[p]);
                  inc(p)
                end
          end
      end;
    writeln(s)
end.
```

Limbajul C/C++:

```
#include <stdio.h>
#include <string.h>
```

```
int literamica(char c){
  if(c>='a' && c<='z')
    return 1;
  return 0;
}
int litera(char c){
  if(c>='a' && c<='z')
    return 1;
  if(c>='A' && c<='Z')
    return 1;
  return 0;
}
int vocala(char c){
  char voc[]="AEIOUaeiou";
  if(strchr(voc,c)!=NULL)
    return 1;
  return 0;
}
char majuscula(char c){
  return c-'a'+'A';
}
int main(){
  int n,i,p,u;
  char s[101];
  gets(s);
  for(i=0;s[i]!=0;i++)
    if(litera(s[i])){
      if(i==0||!litera(s[i-1]))
        p=i;
      if(!litera(s[i+1])){
        u=i;
        if(vocala(s[p])&&vocala(s[u]))
          while(p<=u){
            if(literamica(s[p]))
              s[p]=majuscula(s[p]);
            p++;
          }
      }
    }
  puts(s);
  return 0;
}
```

Atât în limbajul **Pascal**, cât și în **C/C++** putem proceda în felul următor: parcurgem șirul de caractere **s** și, pentru fiecare cuvânt, reținem în variabila de tip întreg **p** poziția la care începe fiecare cuvânt, și în variabila **u,** poziția la care se termină.
Vom prelucra doar caracterele de tip literă. Condiția de a ne afla la începutul unui cuvânt este ca înaintea caracterului curent să nu se afle o literă, sau caracterul curent să fie primul din întregul șir. Condiția de a ne afla pe ultima poziție din cuvânt este ca după caracterul curent să nu se mai afle o literă sau să fim la sfârșitul șirului de caractere. Practic, interpretăm cuvintele ca fiind secvențe continue formate exclusiv din litere.
Apoi, în cazul în care cuvântul curent este unul dintre cele care trebuie prelucrate, facem acest lucru (în cazul nostru transformăm în majuscule) de la **p** la **u**. Această strategie generală poate fi adaptată aproape la orice problemă care cere prelucrarea cuvintelor. Nu depinde nici de numărul de separatori, nici de tipul lor, nici de existența unor restricții legate de începutul sau sfârșitul textului.

Se acordă **1 punct** pentru declararea corectă a variabilelor, **1 punct** pentru citirea datelor (trebuie avut în vedere faptul că șirul de caractere poate conține spații), **1 punct** pentru afișarea textului modificat.
Se acordă **2 puncte** pentru parcurgerea șirului de caractere, **2 puncte** pentru identificarea cuvintelor care trebuie transformate și **2 puncte** pentru modificarea acestora (aceste 2 puncte se acordă numai dacă șirul de caractere este transformat în memorie, conform cerinței, nu și pentru varianta în care rezultatul cerut este doar afișat).
Pentru corectitudinea globală a programului se acordă **1 punct** – în total **10 puncte.**

4. **b)** Pentru a reține șirul numerelor citite de pe cea de-a doua linie a fișierului vom utiliza un vector **`v`**, care va fi ordonat crescător (conform precizării din enunț).

Apoi vom parcurge vectorul de la stânga la dreapta și pentru fiecare indice **`i`** vom determina cel mai mare **`j`** cu proprietatea că **`v[i]+v[j]≤x`**. Având în vedere faptul că vectorul **`v`** este ordonat crescător, rezultă că pentru orice **`k`** natural, cu **i<k≤j**, avem **v[i]+v[k]≤x**, deci numărul de perechi cu proprietatea cerută, pentru care **`v[i]`** este primul termen (mai mic) este **`j-i`**; acesta se va aduna la variabila **`nr`**, în care vom păstra rezultatul final.

Ne vom opri atunci când **`i`** va ajunge mai mare sau egal cu **`j`**, pentru că de la această valoare a lui **`i`** încolo nu va trebui să mai adunăm nimic la **`nr`** (**`v[i]`** a devenit prea mare).

Eficiența algoritmului, ca timp de executare, constă în faptul că **`j`** va face o singură parcurgere de la stânga la dreapta. Ținând cont de faptul că **`v`** este ordonat crescător, observăm că pe măsură ce **`i`** crește, **`j`** trebuie să scadă (sau eventual să rămână pe loc). Deci, atât **`i`**, cât și **`j`** fac cel mult o parcurgere completă a șirului.

Pentru o descriere coerentă a metodei se acordă **1 punct**, iar pentru justificarea eficienței, încă **1 punct** – în total **2 puncte.**

Menționăm că descrierea coerentă a problemei nu presupune să povestim programul, ci ideea de prelucrare. În acest sens e mai bine să explicăm că "parcurgem", "numărăm", "căutăm primul element care …" etc. decât să dăm detalii tehnice cum ar fi "folosim un **`for`**", "incrementăm variabila întreagă **`ț`**…", etc.

a) **Limbajul Pascal**

```
var n,x,i,j,nr:longint;
    v:array[1..100000] of longint;
    fin:text;

begin
  assign(fin,'bac.in');
  reset(fin);
  readln(fin,n,x);
  for i:=1 to n do
    read(fin,v[i]);
  close(fin);
  nr:=0;
  i:=1;
  j:=n;
```

```
  while i<j do
    begin
      while (j>i)and(v[j]+v[i]>x) do
        dec(j);
      nr:=nr+j-i;
      inc(i)
    end;
  write(nr)
end.
```

Limbajul C/C++

```
#include <stdio.h>
int main()
{
  FILE *fin;
  int n,x,v[100000],i,j,nr;
  fin=fopen("bac.in","r");
  fscanf(fin,"%d%d",&n,&x);
  for(i=0;i<n;i++)
    fscanf(fin,"%d",&v[i]);
  fclose(fin);
  nr=0;
  i=0;
  j=n-1;
  while(i<j)
  {
    while(j>i&&v[j]+v[i]>x)
      j--;
    nr+=j-i;
    i++;
  }
  printf("%d",nr);
  return 0;
}
```

Se acordă **1 punct** pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă **1 punct** pentru citirea **tuturor** numerelor din fişier, **3 puncte** pentru un algoritm principial corect, și **1 punct** pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă **2 puncte** pentru alegerea unui algoritm eficient ca timp de executare (O(**n**)) – în total **8 puncte**.

Menționăm că prin *algoritm principial* corect se înțelege un algoritm care, în intenție, urmărește să rezolve corect problema, deși poate să prezinte scăpări logice sau confuzii.

VARIANTA 1

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Indicaţi expresia care are valoarea **true/1** dacă şi numai dacă valoarea reținută de variabila **n** **NU** aparține intervalului [**a**,**b**], unde **a<b**.

	Limbajul Pascal	Limbajul C/C++
a.	(n<a) and (n>b)	n<a && n>b
b.	(n<=a) or (n>=b)	n<=a || n>=b
c.	not ((n>a) and (n<b))	!(n>a && n<b)
d.	n<a or n>b	n<a || n>b

2. Pentru subprogramul **f** cu definiția următoare, indicați ce se va afișa în urma apelului **f(2,4)**?

Limbajul Pascal

```
procedure f(x,y:integer);
begin
  if x<y then
    begin
      inc(x);
      f(x,y);
      write(x+y)
    end;
  write(x,y)
end;
```

Limbajul C/C++

```
void f(int x, int y){
  if(x<y){
    x++;
    f(x,y);
    printf("%d",x+y);
          | cout<<x+y;
  }
  printf("%d%d",x,y);
        | cout<<x<<y;
}
```

a. 73484444	b. 44844734	c. 78444434	d. 34844744

3. Utilizând metoda backtracking se generează, în ordinea crescătoare a valorilor toate numerele formate din exact **n** cifre care au suma cifrelor **s**. Astfel, pentru **n=2**, și **s=4** se obţin, în ordine, numerele: **13**, **22**, **31**, **40**. Folosind aceeaşi metodă, se generează numerele formate din trei cifre, având suma cifrelor **5**.

Care va fi al șaselea număr generat?

a. 203	b. 212	c. 140	d. 230

4. Se consideră un graf neorientat **G** cu **5** vârfuri și **8** muchii. Care dintre următoarele afirmații referitoare la **G** este cu siguranță falsă?

a.	G este conex
b.	G este aciclic (are cel puțin un ciclu)

c.	`G este bipartit`
d.	`G are vârfuri cu grad impar`

5. Se consideră tipul de date **`punct`**, capabil să memoreze abscisa şi ordonata unui punct din plan şi tipul de date **`segment`**, capabil să memoreze două puncte distincte reprezentând extremităţile unui segment din plan.

Limbajul Pascal

```
type punct=record
             x,y:real
           end;
type segment=record
               A,B:punct
             end;
var s:segment;
```

Limbajul C/C++

```
typedef struct {
          float x,y;
} punct;
typedef struct {
         punct A,B;
} segment;
segment s;
```

Care dintre următoarele expresii are ca valoare abscisa mijlocului segmentului corespunzător variabilei **`s`** de tipul **`segment`**?

	Limbajul Pascal	Limbajul C/C++
a.	`(A.x+B.x)/2`	`(A.x+B.x)/2`
b.	`s.(A.x+B.y)/2`	`s.(A.x+B.y)/2`
c.	`(A.s.x+B.s.x)/2`	`(A.s.x+B.s.x)/2`
d.	`(s.A.x+s.B.x)/2`	`(s.A.x+s.B.x)/2`

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **`x%y`** restul împărţirii numărului întreg **`x`** la numărul întreg nenul **`y`**.

```
citeşte n,k (numere naturale nenule)
s←0
┌pentru i←1,n execută
│ ┌dacă i%k=0 atunci
│ │    s←s+i
│ └■
└■
scrie s
```

a) Ce se va afişa dacă se citesc valorile `20` şi `3`? **(6p.)**

b) Pentru **`n=25`**, scrieţi o valoare care trebuie să fie introdusă pentru **`k`** astfel încât rezultatul afişat să fie `60`. **(6p.)**

c)	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, care să nu conţină structuri repetitive **(6p.)**
d)	Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**
2.	Se consideră următoarele declarări de variabile:

Limbajul Pascal	Limbajul C/C++
`var s:string[100];` `aux:char;` `p,u:integer;`	`char s[101],aux;` `int p,u;`

Variabila **s** memorează o propoziție. Caracterele pe care le conține **s** pot fi litere mici și mari ale alfabetului englez, caractere spațiu, virgulă sau punct. Cuvintele pot fi separate prin unul sau mai multe caractere spațiu sau virgulă, șirul de caractere începe cu o literă mare și se termină cu un caracter punct. Scrieți **doar** secvența de instrucțiuni care transformă șirul dat astfel încât ordinea literelor primului cuvânt să fie inversată. De exemplu, dacă inițial **s** reține textul: **Gigel are mere, pere si gutui.** În urma executării secvenței de instrucțiuni cerute **s** va deveni: **legiG are mere, pere si gutui.** **(6p.)**

3. Cu ce trebuie completate punctele de suspensie din secvenţa de instrucţiuni de mai jos astfel încât în urma executării acesteia variabila **niv** să rețină nivelul pe care se află **nod** (numărul de muchii ale lanțului elementar care unește **nod** cu rădăcina arborelui)? Arborele cu rădăcină din care face parte **nod** este codificat prin vectorul de tați **t** definit prin:

- **t[i] =** tatăl nodului **i**, dacă **i** nu este rădăcina arborelui
- **t[i] = 0** dacă **i** este rădăcina arborelui. **(6p.)**

Limbajul Pascal	Limbajul C/C++
`niv := 0;` `while t[nod] <> 0 do` `  begin` `    inc(niv);` `    ...` `  end`	`niv = 0;` `while (t[nod] != 0)` `{` `  niv++;` `  ...;` `}`

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1.) Scrieţi definiţia completă a unui subprogram **desc**, cu patru parametri:

- **n**, prin intermediul căruia primeşte un număr natural format din cel mult **9** cifre
- **d**, prin intermediul căruia primeşte un alt număr natural format din cel mult **9** cifre
- **p**, prin intermediul căruia furnizează cea mai mare putere a lui **d** care îl divide pe **n**
- **e**, prin intermediul căruia furnizează exponentul corespunzător lui **p**

	De exemplu, dacă **n=48** și **d=2**, atunci prin intermediul parametrilor **p** și **e** vor fi furnizate rezultatele **16** și respectiv **4**, pentru că 2^4=**16** îl divide pe **48**, iar 2^5=**32** nu îl divide pe **48**. **(4p.)** b) Scrieți un program **Pascal/C/C++** care, folosind apeluri utile ale subprogramului **desc** descris la punctul **a)**, determină și afișează pe ecran descompunerea în factori primi a unui număr natural **n** (**2≤n≤1000000**) citit de la tastatură. Se vor afișa pe ecran divizorii primi ai lui **n** și puterile la care aceștia apar în descompunerea lui **n**, toate numerele fiind scrise pe o singură linie, separate prin câte un spațiu. **Exemplu:** dacă de la tastatură se citește **n=18** (care are descompunerea în factori primi 2^1*3^2), atunci programul va afișa pe ecran **2 1 3 2** iar dacă se citește **x=360** (care are descompunerea în factori primi $2^3*3^2*5^1$), se va afișa **2 3 3 2 5 1** **(6p.)**
2.	Scrieți un program care citește de la tastatură două numere naturale **m** și **n** (**2≤m≤100, 2≤n≤100**) și **m** linii a câte **n** numere, valorile de pe fiecare linie în parte fiind separate prin câte un spațiu, reprezentând componentele unei matrice cu **m** linii și **n** coloane. Elementele matricei sunt numere întregi având cel mult **4** cifre fiecare. Programul va transforma matricea astfel încât fiecare linie să aibă pe prima poziție cea mai mare componentă a sa, iar pe ultima poziție pe cea mai mică, elementele liniei rămânând însă aceleași. Noua matrice va fi afișată pe ecran pe linii diferite, cu spaţii între elementele de pe fiecare linie. De exemplu, dacă se citesc de la tastatură **m=3**, **n=5**, și matricea de mai jos **10 5 18 2 14** **21 4 12 8 9** **5 11 51 100 8** atunci programul ar putea construi și afișa matricea (aceasta nefiind singura variantă corectă): **18 5 10 14 2** **21 12 8 9 4** **100 11 51 8 5** **(10p.)**
3.	Fişierul text **bac.in** conţine pe prima linie cel mult **1000000** de numere naturale de cel mult **5** cifre fiecare, separate prin câte un spațiu, **ordonate strict crescător**. a) Scrieţi un program care, citește de la tastatură un număr natural **a** (**0≤a≤10000**) și, folosind un algoritm eficient atât din punctul de vedere al timpului de executare, cât și al spațiului de memorie utilizat determină şi afişează pe ecran, pe o singură linie, separate prin câte un spaţiu, numerele din fișierul **bac.in** care sunt puteri ale lui **a**. **(8p.)** **Exemplu:** dacă fişierul **bac.in** are conţinutul: **1 2 3 4 6 7 8 9 11 15 16 21 22 128 129** și de la tastatură se introduce **a=2**, atunci programul va afișa pe ecran: **1 2 4 8 16 128** b) Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

VARIANTA 12 - *Rezolvare*

Subiectul 1

1.	Varianta **a** este incorectă. Niciun număr `n` nu poate îndeplini simultan condițiile `n<a` și `n>b`, în condițiile în care `a<b`. Varianta **b** este de asemenea incorectă, fiind echivalentă cu neapartenența lui `n` la `(a,b)`, adică la intervalul deschis, în locul celui închis, cerut în enunț. Varianta **c** este greșită, expresia logică fiind de fapt echivalentă cu cea de la punctul **b** (neapartenență la intervalul deschis în locul celui închis). Varianta **d** este corectă. Pentru precizarea răspunsului **d)** se acordă **4p.**
2.	În apelul `f(2,4)` va fi incrementat `x` și se va face apelul recursiv `f(3,4)`. În cadrul acestuia va fi din nou incrementat `x` și se va apela `f(4,4)`. În `f(4,4)` se va afișa `44`. Apoi se va reveni în apelul `f(3,4)` și se va afișa `844` (`x` a fost deja incrementat). Se revine apoi în apelul `f(2,4)` și se afișează `734`, pentru că și aici `x` a fost modificat, devenind `3` înainte de apelul recursiv. Deci se va afișa `44844734`. Pentru răspunsul **b)** se acordă **4p**.
3.	Ne putem folosi de rezultatele afișate pentru `n=2` și `s=4`. Primele numere de `3` cifre și suma acestora `5` vor avea pe prima poziție `1`. Numerele cu `1` pe prima poziție, care au `3` cifre și suma `5` se obțin din numerele cu `2` cifre care au suma cifrelor `4`, inserând pe prima poziție `1` (`113`,`122`,`131`,`140`). Există o excepție de la această regulă: numărul `104` (primul generat), pentru că `04` nu este un număr valid cu `2` cifre. Prin urmare, există `5` numere cu proprietățile din enunț care au pe prima poziție `1`. Al șaselea număr este de fapt primul număr care începe cu `2`, are `3` cifre și suma cifrelor `5`. Acesta este `203`. Pentru răspunsul **a)** se acordă **4p**.
4.	Un graf neorientat cu `5` vârfuri, care nu este conex poate avea cel mult `6` muchii (păstrăm un vârf izolat și luăm toate muchiile posibile care nu îl au ca extremitate). În cazul nostru, graful are `8` muchii, deci varianta **a** este incorectă (afirmația „**G** este conex” e cu siguranță **adevărată**). Un graf neorientat aciclic cu `5` vârfuri poate avea cel mult `4` muchii. Cum graful G are `8` muchii, rezultă că afirmația „**G** este ciclic” va fi cu siguranță adevărată, deci varianta **b)** este incorectă. Numărul maxim de muchii pe care le poate avea un graf bipartit cu `5` vârfuri este `6`. La această valoare se ajunge atunci când **G** este $K_{2,3}$ adică un graf bipartit complet cu `2` vârfuri în prima mulțime și `3` în cea de-a doua. Prin urmare, **G** NU este cu siguranță bipartit, răspunsul corect fiind **c)**. Oricum am alege cele `8` muchii, **G** va avea vârfuri cu grad par și vârfuri cu grad impar, prin urmare varianta **d)** nu este corectă. Pentru precizarea răspunsului **c)** se acordă **4p**.

5.		Varianta **d)** este cea corectă. Pentru răspunsul **d)** se acordă **4p**.

Subiectul 2

1.	**a)**	Algoritmul prezentat determină și afişează suma multiplilor lui `k` mai mici sau egali cu **`n`**. Pentru **`n=20`** și **`k=3`**, aceștia sunt: **`3,6,9,12,15,18`**. Deci răspunsul corect este **`63`**. Pentru precizarea valorii **`63`** se acordă **6p**.
	b)	Cum suma cerută, **`60`**, are ca divizori numerele `1,2,3,4,5,6,10,12,15,20,30,60` și termenii care o compun sunt multipli ai lui **`k`**, rezultă că putem restrânge aria de căutare a acestuia la **`1,2,3,4,5,6,10,12,15,20`** (valorile mai mici sau egale cu **`25`**). Luând fiecare variantă posibilă în parte, dar adunând multiplii începând cu cei mai mari, vom scoate din discuție repede toate opțiunile greșite, ajungând la răspunsul corect, **`k=6`** (multiplii lui **`6`** mai mici sau egali cu **`25`** sunt: **`6,12,18,24`**, care au suma **`60`**). O abordare mai elegantă și care totodată presupune efectuarea unui număr mai mic de calcule se bazează pe găsirea unei formule matematice. Multiplii lui **`k`** mai mici sau egali cu **`n`** sunt numere de forma **`k*1,k*2, ...k*m`**, unde **`m=[n/k]`** (parte întreagă din **`n/k`**). Suma lor este **`k*(1+2+...+m) = k*m*(m+1)/2`**. Putem să calculăm sume de forma **`m*(m+1)/2`** și să ne oprim la prima care corespunde unei valori valide a lui **`k`**, în cazul nostru **`m=4`**. Pentru valori mai mici ca **`4`** ale lui **`m`** se obțin de asemenea sume care îl divid pe 60, dar în aceste cazuri **`k*m`** ar depăși **`25`**. Pentru precizarea valorii **`6`**, se acordă **6p**.
	c)	Rezultatul cerut este o sumă de forma **`k*1 + k*2 + ... + k*m = k*(1+2+...+m) = k*m*(m+1)/2`**, unde **`m=[n/k]`**. Prin urmare, un algoritm echivalent, care nu folosește instrucțiuni repetitive este următorul: `citeşte n,k (numere naturale nenule)` `m← [n/k]` `scrie k*m*(m+1)/2` Pentru alegerea unui algoritm echivalent cu cel dat se acordă **2p.**, iar pentru obținerea rezultatului fără utilizarea niciunei structuri repetitive se acordă încă **2p**. – în total **4p**.
	d)	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, experiențe care au cimentat cunoștințele privind structura programelor, declararea variabilelor,

		sintaxa instrucțiunilor programului, regulile de scriere a expresiilor etc. Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și de decizie se acordă câte **1p.**, pentru cele două instrucţiuni repetitive se acordă **3p.**, iar pentru cele **4** atribuiri se acordă **2p.** Pentru structura corectă a programului se mai acordă **1p.**, în total **10p**.

2. Se determină mai întâi poziția din **s** la care se termină primul cuvânt (ne folosim de faptul că în enunț ni se garantează că textul începe cu o literă). Apoi, în cea de-a doua buclă **while** se inversează ordinea literelor primului cuvânt prin interschimbări succesive ale lui **s[p]** și **s[u]**.

Limbajul Pascal

```
p:=1;
u:=1;
while(s[u]<>' ')and(s[u]<>',')and(s[u]<>'.') do
  inc(u);
dec(u);
while p<u do
  begin
    aux:=s[p];
    s[p]:=s[u];
    s[u]:=aux;
    inc(p);
    dec(u)
  end;
```

Limbajul C/C++

```
p=u=0;
while(s[u]!=' '&&s[u]!=','&&s[u]!='.')
  u++;
u--;
while(p<u)
{
  aux=s[p];
  s[p]=s[u];
  s[u]=aux;
  p++;
  u--;
}
```

Pentru determinarea corectă a poziției de sfârșit a (lungimii) primului cuvânt se acordă **2p.**, pentru inversarea ordinii literelor acestuia se acordă **3p**., iar pentru corectitudinea sintactică a secvenței se mai acordă **1p.** – în total **6p**.

3. Pentru a calcula nivelul pe care se află **nod** e necesară parcurgerea drumului lanțului (unic) elementar dintre **nod** și rădăcina arborelui (rădăcina este considerată a fi pe nivelul 0). Acest lanț e parcurs prin înlocuirea succesivă a lui **nod** cu tatăl acestuia (**t[nod]**), până când se ajunge la rădăcină. Numărul pașilor parcurși este rezultatul dorit (**niv**).

Secvența de cod devine:

Limbajul Pascal	Limbajul C/C++
`niv := 0;` `while t[nod] <> 0 do` `  begin` `    inc(niv);` `    nod := t[nod]` `  end`	`niv = 0;` `while (t[nod] != 0)` `{` `  niv++;` `  nod = t[nod];` `}`

Se acordă **2p.** pentru modificarea variabilei **`nod`**, **2p.** pentru folosirea vectorului de tați **`t`** în vederea parcurgerii lanțului către rădăcină și **2p**. pentru corectitudinea de ansamblu a instrucțiunii de atribuire:

`nod := t[nod]` **(Pascal)**, respectiv **`nod=t[nod];`** **(C/C++)**.

Subiectul 3

1. **a)** Procedura/funcția **`desc`** are conform enunțului **4** parametri: **`n,d,p`** și **`e`**. Dintre aceștia ultimii doi trebuie să fie referințe pentru ca prin intermediul lor să fie furnizate rezultatele dorite. Nu sunt necesare variabile locale. Pur și simplu se inițializează **`p`** cu **`1`** (fiind de fapt un produs) și **`e`** cu **`0`** (**e** este un contor). Cât timp **`n`** mai poate fi împărțit exact la **`d`** se face împărțirea și se actualizează cele două rezultate, **`p`** și **`e`**.

<u>Limbajul Pascal</u>

```
procedure desc(n, d: longint; var p, e: longint);
begin
  p := 1;
  e := 0;
  while n mod d = 0 do
    begin
      inc(e);
      p := p * d;
      n := n div d
    end
end;
```

<u>Limbajul C/C++</u>

```
void desc(int n, int d, int&p, int&e)
{
  p = 1;
  e = 0;
  while (n % d == 0)
  {
    e++;
    p = p * d;
    n = n / d;
  }
}
```

Pentru respectarea structurii antetului (**`procedure`**/**`void`**), declararea corectă a parametrilor de intare **`n`** și **`d`** și a parametrilor de ieșire **`p`** și **`e`** se acordă **1p.**, pentru inițializarea adecvată a lui **`p`** și respectiv **`e`** se acordă **1p.**, iar pentru structura repetitivă în cadrul căreia sunt calculate rezultatele încă **2p.** – în total **4 puncte**.

b) Pentru a obține descompunerea în factori primi a lui **n** trebuie să găsim divizorii primi ai acestuia. O observație esențială este aceea că dacă de fiecare dată când găsim un divizor îl împărțim efectiv pe **n** la acesta de câte ori este posibil, atunci vom găsi doar divizorii primi, nefiind necesară și verificarea acestei proprietăți. Pornind de la această observație putem scrie o buclă **while** în cadrul căreia, apelând subprogramul **desc** determinăm descompunerea în factori primi.

Limbajul Pascal

```
d := 2;
while n <> 1 do
 begin
   if n mod d = 0 then
     begin
       desc(n,d,p,e);
       write(d,' ',e,' ');
       n := n div p
     end;
   inc(d)
 end
```

Limbajul C/C++

```
d = 2;
while (n != 1){
 if (n % d == 0) {
   desc(n, d, p, e);
   printf("%d %d ",d,e);
   n=n/p;
 }
 d++;
}
```

În cazul cel mai defavorabil (atunci când **n** este prim) bucla **while** de mai sus face **n** pași. Pornind de la observația că **n** nu poate avea mai mult de un divizor prim mai mare ca radicalul său, putem optimiza algoritmul astfel încât să nu avem mai mult de radical din **n** iterații în buclă. Lipsa acestei optimizări nu va atrage însă depunctarea.

Limbajul Pascal

```
d := 2;
while d * d <= n do
 begin
   if n mod d = 0 then
     begin
       desc(n, d, p, e);
       write(d,' ',e,' ');
       n := n div p
     end;
   inc(d)
 end;
if n <> 1 then
  write(n,' ',1)
```

Limbajul C/C++

```
d = 2;
while (d * d <= n){
 if (n % d == 0){
   desc(n, d, p, e);
   printf("%d %d ",d,e);
   n = n / p;
 }
 d++;
}
if (n != 1)
  printf("%d 1", n);
```

Pentru declararea tuturor variabilelor se acordă **1p.**, pentru citirea lui **n** și inițializarea lui **d** se acordă încă **1p.**, pentru găsirea tuturor divizorilor primi ai lui **n** se acordă **1p.** (se acordă numai în cazul în care nu sunt afișați și alți divizori), pentru apelul corect și util al subprogramului de la punctul **a)** se acordă **1p.+1p.**, pentru corectitudinea globală a programului se mai acordă **1p.** – în total **6 puncte**.

2. Programul trebuie să citească matricea, să o modifice (prelucreze) conform cerinței și apoi să o afișeze. Primele două operații pot fi realizate simultan, dar (mai ales în condițiile unui examen scris) este recomandat să fie implementate separat, pentru a reduce riscurile unor greșeli.

Nu vom insista asupra citirii și afișării matricei. Prelucrarea în schimb necesită o oarecare atenție. Mai întâi observăm că în urma parcurgerii fiecărei linii vom avea nevoie de pozițiile celui mai mic (**pmin**) și respectiv celui mai mare element (**pmax**) și nu de valorile acestora. Apoi, prin interschimbări, trebuie adus cel mai mare element al liniei pe prima poziție a acesteia, iar cel mai mare pe ultima.

O capcană în care se poate cădea ușor este cazul particular în care cel mai mic element se află pe prima poziție. Într-o astfel de situație aducerea valorii maxime pe prima poziție poate avea ca efect pierderea poziției celui mai mic element. De aceea este necesar ca **pmin** să primească valoarea **pmax** (poziția pe care se va afla cel mai mic element după ce pe prima poziție va fi adus cel mai mare).

Pentru a ușura prelucrarea am folosit o procedură/funcție de interschimbare a doi întregi. Este important ca parametrii formali ai acesteia să fie referințe.

Limbajul Pascal

```
var m, n, i, j, pmin, pmax, aux: integer;
    a: array [1..100,1..100] of integer;

procedure schimb(var x, y:integer);
var aux: integer;
begin
  aux := x;
  x := y;
  y := aux
end;

begin
  {citirea matricei}
  read(m,n);
  for i := 1 to m do
    for j := 1 to n do
      read(a[i,j]);
  {prelucrarea matricei}
  for i := 1 to m do
    begin
      pmin := 1;
      pmax := 1;
      for j := 2 to n do
        begin
          if a[i,j] < a[i,pmin] then
            pmin := j;
          if a[i,j] > a[i,pmax] then
            pmax := j;
        end;
      if pmin = 1 then
        pmin := pmax;
      schimb(a[i,1], a[i,pmax]);
      schimb(a[i,n], a[i,pmin])
    end;
  {afisarea matricei}
  for I := 1 to m do
    begin
      for j := 1 to n do
        write(a[i,j], ' ');
      writeln
    end
end.
```

Limbajul C/C++

```
#include <stdio.h>

int m, n, a[101][101];

void schimb(int *x, int *y)
{
  //interschimbarea
  int aux = *x;
  *x = *y;
  *y = aux;
}

int main()
{
  int i, j, pmin, pmax, aux;
  //citirea matricei
  scanf("%d%d",&m,&n);
  for (i = 1; I <= m; i++)
    for (j = 1; j <= n; j++)
      scanf("%d", &a[i][j]);
  //prelucrarea matricei
  for (i = 1; i <= m; i++)
  {
    pmin = pmax = 1;
    for (j = 2; j<=n; j++)
    {
      if (a[i][j] < a[i][pmin])
        pmin = j;
      if (a[i][j] > a[i][pmax])
        pmax = j;
    }
    if (pmin == 1)
      pmin = pmax;
    schimb(&a[i][1], &a[i][pmax]);
    schimb(&a[i][n], &a[i][pmin]);
  }
  //afisarea matricei
  for (i = 1; i <= m; i++)
  {
    for (j = 1; j <= n; j++)
      printf("%d ", a[i][j]);
    printf("\n");
  }
  return 0;
}
```

Pentru declararea corectă a matricei se acordă **1p.**, pentru citirea matricei se mai acordă **1p.**, pentru parcurgerea liniilor matricei se acordă **1p**., pentru determinarea pozițiilor corespunzătoare valorilor minimă și respectiv maximă, se acordă **1p+1p.**, pentru mutarea valorilor minimă și respectiv maximă pe pozițiile cerute (ultima și respectiv prima) se acordă **1p.+1p.**, pentru tratarea cazului particular în care cel mai mare element al liniei se află pe prima coloană se acordă **1p.**, iar pentru afișarea matricei conform cerinței se acordă **1p**. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă **1p.** – în total **10 puncte**.

4. **b)**

Vom utiliza doar 3 variabile întregi: **a**, în care vom reține valoarea citită de la tastatură, **p** în care vom păstra puterea curentă a lui **a** și **x** în care vom reține pe rând numerele citite din fișierul text.

Bazându-ne pe faptul că numerele din fișier sunt ordonate strict crescător vom compara de fiecare dată ultimul număr citit, **x** cu puterea curentă a lui **a**, reținută în **p**. Avem 3 situații posibile:

a. Dacă **p** și **x** sunt egale, îl vom afișa pe **x**, vom citi un nou număr din fișier și vom trece la următoarea putere a lui **a** (**p←p*a**)

Dacă **p** este mai mic, îl vom actualiza doar pe el.

Dacă **x** este mai mic vom citi o nouă valoare din fișier.

Algoritmul este eficient din punctul de vedere al spațiului de memorie, întrucât nu păstrează la un moment dat decât o singură valoare (ultima) citită din fișier, nefolosind astfel tablouri.

Din punctul de vedere al timpului de executare algoritmul este de asemenea eficient, făcând o singură parcurgere a numerelor din fișier și calculând o singură dată puterile lui **a**.

Pentru o descriere coerentă a metodei se acordă **1p.**, iar pentru justificarea eficienței, încă **1p.** – în total **2p.**

Menționăm că descrierea coerentă a problemei nu presupune să povestim programul, ci ideea de prelucrare. În acest sens e mai bine să explicăm că “parcurgem”, “numărăm”, ”căutăm primul element care …” etc. decât să dăm detalii tehnice cum ar fi ”folosim un **for**”, ”incrementăm variabila întreagă **ț**…” etc.

a)

Limbajul Pascal

```
var fin:text;
    a,p,x:longint;

begin
  assign(fin, 'bac.in');
  reset(fin);
  read(a);
  p := 1;
  read(fin, x);
  while not eof(fin) do
    begin
      if x = p then
        begin
          write(x, ' ');
          read(fin, x);
          p := p * a
        end
      else if x < p then
        read(fin, x)
      else
        p := p * a
    end;
  close(fin)
end.
```

Limbajul C/C++

```
#include <stdio.h>

int main()
{
  FILE *fin;
  int a, p, x;
  fin = fopen("bac.in", "r");
  scanf("%d",&a);
  p = 1;
  fscanf(fin, "%d", &x);
  while (!feof(fin))
  {
    if (x == p)
    {
      printf("%d ", x);
      fscanf(fin, "%d", &x);
      p = p * a;
    }
    else if (x < p)
      fscanf(fin, "%d", &x);
    else
      p = p * a;
  }
  fclose(fin);
  return 0;
}
```

Se acordă **1p.** pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă **1p.** pentru citirea tuturor numerelor din fişier, **2p.** pentru un algoritm principial corect, încă **1p.** pentru determinarea tuturor valorilor cerute și **1p.** pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă **1p.** pentru alegerea unui algoritm eficient ca timp de executare **(O(n))** și **1p.** pentru utilizarea eficientă a memoriei - în total **8p.**

Menționăm că prin algoritm principial corect se înțelege un algoritm care, în intenție, urmărește să rezolve corect problema, deși poate să prezinte scăpări logice sau confuzii.

VARIANTA 13

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieți pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.		
1.	Variabilele x şi y sunt de tip întreg, x = 8, iar y = 6. Care dintre expresiile de mai jos are valoarea FALSE respectiv valoarea 0? **(4p.)**	
	Limbajul Pascal	**Limbajul C/C++**
a.	`3 * x - 4 * y =0`	`3 * x - 4 * y == 0`
b.	`(x+y)div 2 > x mod y + 1`	`(x + y) / 2 > x % y + 1`
c.	`NOT (x div 2 + 2 = y)`	`!(x % 2 + 2 == y)`
d.	`x - y + 3 <> 0`	`x - y + 3 != 0`
2.	Fiind dată declaraţia următoare, afişaţi numele şi iniţiala prenumelui tuturor elevilor care au nota mai mare decât **7**. **(4p)**	
	Limbajul Pascal	**Limbajul C/C++**
	`type elev=record` `nume, prenume: string[20]` `nota: integer;` `End;` `var e:elev;`	`struct elev` `{` `char nume[20], prenume[20];` `int nota;` `} ;` `elev e;`
a.	if(nota.e>7) then write(nume.e, prenume[1].e);	if(nota.e>7) cout<<nume.e<<prenume[0].e);/ printf(”%s%c”,nume.e, prenume[0].e);
b.	if(e.nota >7) then write(e.nume, e.prenume[2]);	if(e.nota >7) cout<< e.nume <<prenume[1].e);/ printf(”%s%c” ,e.nume, e.prenume[1]);
c.	if(e.nota >7) then write(e.nume, e.prenume[1]);	if(e.nota >7) cout<< e.nume <<e.prenume[0]);/ printf(”%s%c” ,e.nume, e.prenume[0]);
d.	if(e.nota >7) then write(e.nume, e.prenume);	if(e.nota >7) cout<< e.nume <<e.prenume);/ printf(”%s%c” ,e.nume, e.prenume);

3. Ce se va afişa în urma executării următoarei secvenţe de **`instrucţiuni`**? **(4p.)**

Limbajul Pascal

```
var t,q:string[20];
    a,i:integer;
begin
t:='informatica';
q:='aeiouAEIOU';
a:=1;
while (a=1) do
       for i:=1 to length(t) do
begin
            a:=0;
         if (pos(t[i],q)<>0)  then
          begin
           delete (t,i,1);
           a:=1;
          end;
        end;
write(t);
end.
```

Limbajul C/C++

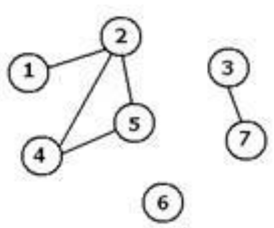
```
char t[]="Informatica",
q[]="aeiouAEIOU";
 int  a=1;
  while(a)
 for(int i=0;i<strlen(t);i++)
 {
  a=0;
  if (strchr(q,t[i])!=0)
   {strcpy(t+i,t+i+1); a=1;}
  }
  cout<<t; | printf("%s",t);
}
```

a. ioaa	b. nfrmtc	c. nomti	d. ifraia

4. Utilizând metoda backtracking se generează toate cuvintele formate din trei vocale litere mici. Cuvintele sunt generate în ordine lexicografică. Primele trei soluţii sunt: **aaa**, **aae** şi **aai**. Afişaţi antepenultima soluţie. (**4p**.)

a. uuo	b. uui	c. uou	d. uuu

5. Pentru graful din imagine scrieţi numărul maxim de componente conexe care se pot obţine eliminând 2 muchii.



(4p.)

a. **4**	b. **5**	c. 3	d. 6

Subiectul 2 (40p)

<table>
<tr><td colspan="3">Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.</td></tr>
<tr><td>1.</td><td colspan="2">Se consideră algoritmul următor reprezentat în pseudocod:
S-a notat cu n%x restul împărţirii numărului întreg n la numărul întreg nenul x şi cu [a] partea întreagă a numărului real a.</td></tr>
<tr><td></td><td colspan="2"><pre>
citeşte n (număr natural)
 p<-0
┌pentru i←1,n execută
| y←-1
| citeşte x (număr natural)
| ┌cât timp x≠0 execută
| | ┌dacă y<x%10 si x%2=0 atunci
| | | y←x%10
| | └■
| | x<-[x/10]
| └■
| ┌dacă y≠-1 atunci
| | p← p*10+y
| └■
└■
scrie p
</pre></td></tr>
<tr><td>a.</td><td colspan="2">Ce valoare va fi afişată dacă se citesc în ordine valorile 5 123 12345 175 301 4552? (6p.)</td></tr>
<tr><td>b.</td><td colspan="2">Scrieţi un set de numere întregi astfel încât pentru n=4 să se afişeze valoarea 286. (6p.)</td></tr>
<tr><td>c.</td><td colspan="2">Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura pentru... cu structura cât timp...execută. (6p.)</td></tr>
<tr><td>d.</td><td colspan="2">Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. (10p.)</td></tr>
<tr><td>2.</td><td>Se consideră graful neorientat cu 8 vârfuri numerotate de la 1 la 8, dat prin listele de adiacenţă de alături:

Să se determine din câte subgrafuri conexe şi aciclice sunt în graful dat. (6p.)</td><td><pre>
1: 2
2: 1,3
3: 2
4: 5
5: 4,6
6: 5
7: 8
8: 7
</pre></td></tr>
</table>

3.	Se consideră subprogramul **E** cu definiţia următoare:	
	Limbajul Pascal	**Limbajul C/C++**
	procedure E(a:byte); begin if(a<9)then begin write(a mod 10); E(a div 10); write(a mod 10); end; end;	void E(int a) { if(a>0) { printf("%d",a%10); / cout<<a%10; E(a/10); printf("%d",a%10); /cout<<a%10; } }
	Ce se va afişa în urma apelului E(175)? **(6p.)**	

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Se citeşte **n** număr natural (n<=10), apoi se citeşte o matrice formată din n*n elemente numere întregi de maxim opt cifre. Se citeşte un număr natural (k<=n). Să se şteargă din matrice coloana **k**. Matricea va fi afişată linie cu linie, elementele de pe fiecare linie vor fi separate prin câte un spaţiu. De exemplu, dacă se citesc de la tastatură **n=4**, **k=2** şi matricea: 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 atunci matricea rezultat va fi: 1 3 4 1 3 4 1 3 4 1 3 4 **(10p.)**

2.	**a)** Scrieţi funcţia număr care primeşte prin intermediul parametrului **x** un număr întreg (**10<=x**) şi returnează prin intermediul parametrului **y** numărul obţinut din prima şi ultima cifră a lui **x**. Exemplu: **x=1234** atunci **y=14** **(5p)**
	b) Fişierul bac.in conţine cel mult 1000000 de numere întregi de maxim patru cifre. Folosind apeluri utile ale funcţiei număr afişati de câte ori apare fiecare număr de două cifre în acest fişier. Numerele vor fi afişate în ordine crescătoare a valorii. Afişarea numărului de două cifre şi a numărului de apariţii ale acestuia se va face pe câte o linie. Cele doua valori de pe linie vor fi separate de un spaţiu. **(12p)** *Exemplu*: bac.in conţine numerele: 3 123 24 45 126 24 88 345 37 45 45 678 33 se va afişa: 24 2 33 1 37 1 45 3 88 1
	c) Explicaţi succint, în limbaj natural, metoda utilizată la punctul b), justificând eficienţa acesteia. **(3p)**

Observaţii:

- Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
- Timpul efectiv de lucru este de 3 ore.
- În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
- În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

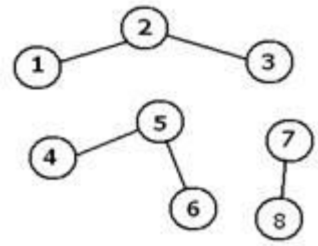
VARIANTA 13 - *Rezolvare*

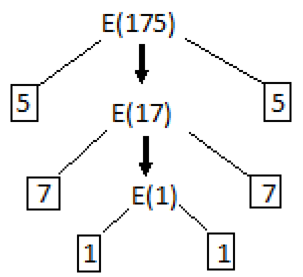
Subiectul 1 - Rezolvare

1.	Pentru răspunsul corect **d)** se acordă **4p**.
2.	Pentru răspunsul corect **c)** se acordă **4 p.**
3.	Şterge toate vocalele din ”Informatica”. Pentru răspunsul corect **b)** ”**nfrmtc**” se acordă **4p.**
4.	Pentru răspunsul corect **b), uui**, se acordă **4p**
5.	Pentru răspunsul **b)** se acordă **4p**. Dacă eliminăm muchiile **[1,2]** şi **[3,7]** obţinem numărul maxim de **5** componente conexe. Pentru oricare altă valoare scrisă se acordă **0** puncte.

Subiectul 2 – Rezolvare

1.	**a)**	Algoritmul ia cea mi mare cifră pară din fiecare număr citit şi cu ajutorul ei construieşte un număr nou. **5** reprezintă numărul de numerele pe care le vom citi **2** este cea mai mare cifră pară a numărului **123** **4** este cea mai mare cifră pară a numărului **12345** Numărul **175** nu are cifre pare **2** este cea mai mare cifră pară a numărului **301** **2** este cea mai mare cifră pară a numărului **4552** Pentru precizarea valorii **2404** se acordă **6p**.
	b)	Valorile posibile sunt: **122 315 108 şi 61** Pentru precizarea a **4** valori cu proprietatea că **3** dintre ele au cele mai mici cifre pare **2, 8, 6** în această ordine şi a unei valori care nu are nici o cifră pară se acordă **6p**. Pentru identificarea a cel puţin **3** valori corecte se acordă **3p**. Pentru alte valori precizate se acordă **0p**.
	c)	Înlocuirea unei structuri **pentru …** cu structura **cât timp n>0 execută …** presupune: `citeşte n (număr natural)` `p<-0` `i←1` `┌while i<=n execută` `│ y←-1` `│ citeşte x (număr natural)` `│ ┌cât timp x≠0 execută` `│ │ ┌dacă y<x%10 si x%2=0 atunci` `│ │ │ y←x%10` `│ │ └■`

		`` | | x<-[x/10] | └■ | ┌dacă y≠-1 atunci | | p← p*10+y | └■ | i←i+1 └■ ``` **scrie p** Pentru alegerea unei structuri repetitive se acordă **1p**., pentru trecerea la următorul numar cu instrucţiunea i<-i+1; se acordă **2p**., pentru testarea condiţiei de continuare sau oprire se acordă **2p**, iar pentru scrierea integrală a restului algoritmului se mai acordă **1p**. - în total **6p.**
	d)	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Prin această cerinţă se valorifică experienţele anterioare de implementare şi testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucţiunilor programului, regulile de scriere a expresiilor, etc. Pentru instrucţiunile corecte de: - declarare a variabilelor, se acordă **1p** - citire a datelor, se acordă **1p** - afişare a rezultatului, se acordă **1p** - decizie, se acordă **1p**., pentru instrucţiunea repetitivă se acordă **2p**., iar pentru cele **5** atribuiri se acordă **3p** (fiecare instrucţiune **0.6 p**). Pentru structura corectă a programului se mai acordă **1p**., în total **10p**.
2.	 Pentru răspuns corect **3** se acordă **6 p**. Pentru oricare altă valoare scrisă, se acordă **0** puncte.	
3.	Pentru răspunsul corect **571175** se acordă **6p**. În desenul următor este prezentat mecanismul prin care se execută apelul **E(175)** al subprogramului recursiv **E**. Se vor afişa, în ordine: **571175**	



Subiectul 3 – Rezolvare

1.

Limbajul Pascal

```
var a:array[1..10,1..10]of integer;
     i,n,j,k,p:integer;
begin
 read(n);
 for i:=1 to n do)
     for j:=1 to n do
           read(a[i][j]);
 read(k);
 for j:=k to n-1 do
     for i:=1 to n do
           a[i][j]:=a[i][j+1];
 p:=n-1;
 for i:=1 to n do
     begin
           for j:=1 to p do
                write(a[i][j],' ');
           writeln;
      end;
end.
```

Limbajul C/C++

```
#include<iostream>
using namespace std;
int main (){
 int i,j,k,p,n,a[10][10];
 cin>>n;
 for(i=1;i<=n;i++)
     for(j=1;j<=n;j++)
        cin>>a[i][j];| scanf("%d",a[i][j]);
 cin>>k;  | scanf("%d",&k);
 for(j=k;j<=n-1;j++)
      {for(i=1;i<=n;i++)
             a[i][j]=a[i][j+1];
      }
 p=n-1;
 for(i=1;i<=n;i++)
       { for(j=1;j<=p;j++)
           cout<<a[i][j]<<" "; | printf("%d",a[i][j]);
         cout<<endl; printf("\n"); }
 return 0;
}
```

	Pentru declararea corectă a matricei se acordă **1p**., pentru citirea lui **n** si a matricei se mai acordă **1p**., pentru parcurgerea unei coloane sau linii se acordă 1p., pentru stergerea corectă a unei coloane se acordă **4p**., iar pentru afişarea matricei conform cerinţei se acordă **2p**. În plus, pentru declararea variabilelor simple, structura şi corectitudinea sintactică a programului se mai acordă **1p**. - în total **10p**.

2. **a)** Pentru antet corect se acordă **2p**. (Pentru respectarea structurii antetului (procedure/void), pentru respectarea structurii subprogramului şi a sintaxei limbajului se acordă încă **1p**, pentru determinarea corectă a numărului y se acordă **2p**. În total **5p**.

Limbajul Pascal

```
procedure număr(x:integer, var y:integer)
begin
   y:=x mod 10;
   while x>9 do
         x=x/10;
   y:=x*10+y;
end;
```

Limbajul C/C++

```
void numar(int x, int &y)
{
    y=x%10;
    while (x>9)
        x=x/10;
    y=x*10+y;
}
```

b)

Limbajul Pascal

```
var f:text;
    i,y,x:integer;
       a: array[1..100] of integer;
begin
  assign(f,'bac.in');reset(f);
  for i:=10 to 99 do
    a[i]:=0;
  while not (eof(f)) do
    begin
      read(f,x);
      număr(x,y);
      if (x=y) then a[x]:=a[x]+1;
    end;
  for i:=10 to 99 do
  if a[i]<>0 then
      writeln(i,' ',a[i]);
end.
```

Limbajul C/C++

```
#include
#include
using namespace std;
ifstream f("bac.in");
```

```
int main()
{   int x,y,i, a[100];
    ifstream f("date.in");
    for(i=10;i<=99;i++)
    a[i]=0;
    while (f>>x)
    {    număr(x,y);
         if (x==y) a[x]++;
    }
    for (i=10;i<=99;i++)
         if(a[i]!=0)
         cout<<i<<" "<<a[i]<<endl;
         | printf("%d%c%d\n",i,' ',a[i]);
    return 0;
}
```

Algoritmul de rezolvare propus:

Citim numerele din fișier și folosind apeluiri ale subprogramului număr determinăm elementele de doua cifre.

Cu ajutorul unui vector caracteristic numărăm de câte ori apare fiecare element de două cifre.

Se acordă **2p**. pentru operaţiile cu fişiere (declarare, nume corect şi deschidere pentru citire), încă **2p**. pentru citirea tuturor numerelor din fişier, **3p**. pentru determinarea vectorului caracteristic încă **2p**. pentru determinarea corectă a perechilor pentru fiecare linie, **2p**. pentru afișarea în formatul cerut, **1p**. Pentru corectitudinea formală (declararea variabilelor, structura programului sintaxa instrucțiunilor, etc.).

Total **12p**.

c) Algoritmul de rezolvare propus citește, unul câte unul, numerele din fișier. Se acordă **2p**. pentru alegerea unui algoritm eficient ca timp de executare (O(n) data de parcugerea o singura data a numerelor din fisier, vector de apariții) și **1p** pentru utilizarea eficentă a memoriei (vector cu 90 de elemente). În total **3p**.

VARIANTA 14

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Indicaţi expresia care are valoarea `true/1` dacă şi numai dacă valoarea variabilei întregi pozitive `x` depășește valoarea variabilei întregi pozitive `y` și diferența lor este cel mult 3.

	Limbajul Pascal	Limbajul C/C++
a.	`abs(x-y)<=3`	`abs(x-y)<=3`
b.	`y-x<=3`	`y-x<=3`
c.	`(x=y+3)or(x=y+2)or(x=y+1)`	`(x=y+3)||(x=y+2)||(x=y+1)`
d.	`x-y<=3`	`x-y<=3`

2. Pentru subprogramul `f` definit mai jos, indicați valoarea expresiei `f(325,1)`.

Limbajul Pascal

```
function f(x,p:integer):integer;
begin
      if x = 0 then f:=0
      else
        f:=f(x div 10,p*10)+x*p
end;
```

Limbajul C/C++

```
int f(int x,int p)
{
      if(x==0) return 0;
      return f(x/10,p*10)+x*p;
}
```

a. 360	b. 945	c.10	d. 523

3. Utilizând metoda backtracking se generează, în ordinea crescătoare a valorii,toate numerele formate din exact **n** cifre distincte, în care orice două cifre vecine au parități diferite. Astfel, pentru **n** egal cu **3** se obțin, în ordine, numerele: **123, 125, 127, 129, 143, 145, ...** Care este ultimul număr generat?

a. 896	b. 989	c. 987	d. 789

4. Se consideră arborele dat prin următorul vector "de taţi": `(6,6,2,3,4,0,3,3)`. Ştiind că rădăcina se află pe nivelul 0, stabiliţi care este numărul de noduri aflate pe nivelul 2 al arborelui:

a. 1	b. 2	c. 3	d. 4

5. Se consideră un graf neorientat **G** cu **6** noduri etichetate cu numere distincte de la 1 la 6, graf dat prin listele de adiacenţă alăturate. Care dintre următoarele variante reprezintă o submulțime cu un număr maxim de noduri ce determină un subgraf **G'** al grafului **G** astfel încât suma gradelor nodurilor în **G'** să fie egală cu **0**. **(6p.)**

```
1: 3
2: 3
3: 1 2
4: 6
5: 6
6: 4 5
```

a. 1,2,4	b. 2,6	c. 1,2,4,5	d. 1,2,5

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
Citeşte n,k    (numere naturale nenule)
n1 ←0 ; n2 ←0 ; p ←1
┌cât timp n ≠ 0 execută
│ ┌dacă n mod 2=0 atunci
│ │ n2 ← n2 + n mod 10*p
│ │altfel
│ │ n1 ← n1 + n mod 10*p
│ └■
│ n ← [n/10]
│ p ← p*10
└■
p ← 1
┌pentru i←1,k execută p ← p*10
└■
x ← [n2/p] + n1 mod p
scrie x
```

a) Ce valoare va fi afişată dacă pentru n se citeşte valoarea **123611**, iar pentru k se citeşte valoarea **2**? **(6p.)**

b) Pentru **k=2**, scrieţi **câte** valori naturale distincte, formate din trei cifre fiecare, pot fi citite pentru variabila **n**, astfel încât, pentru fiecare dintre acestea, valoarea afişată de algoritm să fie **0**. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **cât pentru...execută** cu o structură repetitivă de alt tip. **(6p.)**

d) Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. Se consideră structura **punct**, capabilă să memoreze abscisa şi ordonata unui punct din plan, structura **segment**, capabilă să memoreze două puncte distincte reprezentând extremităţile unui segment din plan și variabila **s** de tipul segment:

Limbajul Pascal	Limbajul C/C++
`type punct=record` `        x,y:real` `     end;` `segment=record` `     A,B:punct` `end;` `var s:segment;`	`struct punct {` `     float x,y;` `};` `struct segment {` `     punct A,B;` `};` `segment s;`

	Scrieţi o expresie în limbajul **Pascal/C/C++** care să aibă valoarea **`true/1`** dacă şi numai dacă variabila **`s`** memorează un segment vertical (aflat pe axa Oy sau paralel cu axa Oy)? **(6p.)**
3.	Se consideră două şiruri de caractere `s` şi `t`. Variabila `s` memorează numele unui elev, variabila `t` memorează numele altui elev.

	Limbajul Pascal	Limbajul C/C++
	`var s,t:string[31];`	**`char s[31],t[31];`**

Ştiind că ambele nume sunt formate doar din litere mari ale alfabetului englez şi sunt nume distincte, scrieţi **doar** instrucţiunea sau instrucţiunile care afişează pe ecran numele aceluia dintre cei doi elevi care trebuie scris în catalog înaintea celuilalt. **(6p.)**

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Subprogramul **`factori`** primește prin intermediul parametrului **`n`** un număr natural cu cel mult 6 cifre cifre (**`n>1`**) și construiește tabloul unidimensional **`v`** care conține 2***`nf`** valori: toți factorii din descompunerea în factori primi a numărului **`n`**, în ordine crescătoare, fiecare factor fiind urmat de exponentul la care acesta apare în descompunere. Subprogramul va avea exact trei parametri, **`n`**, **`v`** și **`nf`**, în această ordine.

Scrieţi definiţia completă a subprogramului **`factori`**.

Exemplu: pentru **`n=350`**, **`nf`** va furniza valoarea 3, iar vectorul **`v`** va conține valorile (**`2`**, **`1`**, **`5`**, **`2`**, **`7`**, **`1`**), deoarece numărul 350 se descomune în 3 factori primi: $2^1*5^2*7^1$. **(10p.)**

2. Scrieți un program **`Pascal/C/C++`** care construiește o matrice cu `n` linii și `n` coloane care conţine pe prima coloană, de sus în jos, toate numerele naturale de la `1` la `n`, în ordine crescătoare, pe coloana a doua, de sus în jos, toate numerele naturale de la `n` la `1`, în ordine descrescătoare, pe a treia coloană, de sus în jos, toate numerele naturale de la `1` la `n`, în ordine crescătoare, pe a patra coloană, de sus în jos, toate numerele naturale de la `n` la `1`, în ordine descrescătoare şi aşa mai departe până la coloana `n`. Scrieţi programul care citeşte de la tastatură o valoare naturală **`n`** **`(3<=n<=25)`**, construieşte matricea conform cerinţei şi o afişează pe ecran, pe linii, cu spaţii între elementele de pe fiecare linie. **(10p.)**

De exemplu, dacă se citeşte de la tastatură **`n=5`**, se va construi şi se va afişa matricea:

```
1 5 1 5 1
2 4 2 4 2
3 3 3 3 3
4 2 4 2 4
5 1 5 1 5
```

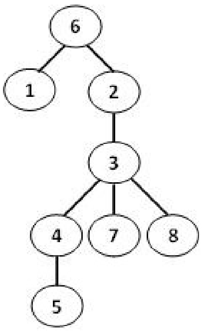
3.	Fişierul text **`bac.in`** conţine un şir cu cel mult **`1000000`** de numere naturale cu valori de cel mult 8 cifre, valori separate prin spații. Se știe că în fişier există cel puțin un număr de o cifră. Numim **secvență** unul, două sau mai multe numere aflate pe poziții consecutive în şir. a) Scrieţi un program Pascal/C/C++ care, utilizând un algoritm eficient din punct de vedere al timpului de executare și al spațiului de memorie utilizat, determină şi afişează pe ecran numerele aparținând ultimei secvențe formate numai din valori cu o singură cifră, în ordinea crescătoare a valorilor din secvență. **(8p.)** **Exemplu:** dacă fişierul **`bac.in`** are conţinutul: **`2 10 4 6 99 55 7 0 2 43 5 7 5 400 81`** atunci se afişează **`5 5 7`** b) Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

⚠ **Observaţii:**

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 14 - *Rezolvare*

Subiectul 1

1.	**`x-y<=3`** este adevărat și în cazul în care **`x`** nu depășește pe **`y`**. Pentru precizarea răspunsului **c)** se acordă **4p**.
2.	Pentru răspunsul **b)** se acordă **4p**.
3.	Pentru răspunsul **c)** se acordă **4p**.
4.	Arborele dat poate fi reprezentat prin figura de mai jos. Nodul 6 (rădăcina) este pe nivelul 0, nodurile 1 și 2 se află pe nivelul 1, iar nodul 3 se află pe nivelul 2.  Pentru răspunsul **a)** se acordă **4p**.
5.	Pentru precizarea răspunsului **c)** se acordă **4p**.

Subiectul 2

1.	**a)**	Se formează numerele **`n1`** și **`n2`** prin anularea cifrelor impare și respectiv pare din scrierea numărului **`n`** și se adună numărul format din ultimele **`k`** cifre ale lui **`n1`** cu numărul format din **`n2`** prin eliminarea ultimelor **`k`** cifre. Pentru precizarea valorii **`217`** se acordă **6p**.
	b)	Din scrierea $\overline{abc}$ a unui astfel de număr se observă că rezultatul afișat este `a'+`$\overline{\texttt{b'c'}}$, unde **`a'`** este chiar **`a`** dacă **`a`** era cifră pară sau 0 dacă **`a`** era cifră impară, **`b'`** este chiar **`b`** dacă **`b`** era cifră impară sau 0 dacă **`b`** era cifră pară, **`c'`** este chiar **`c`** dacă **`c`** era cifră impară sau 0 dacă **`c`** era cifră pară. Pentru ca suma să fie 0, trebuie ca toate cifrele **`a', b', c'`** să aibă valoarea 0, deci obținem produsul cartezian: {1,3,5,7,9} x{0,2,4,6,8} x{0,2,4,6,8} care are 5x5x5 elemente. Pentru enumerarea unor valori corecte (oricâte) se acordă **3p.**, iar pentru precizarea răspunsului **125**, se acordă **6p**.

c) Deoarece valoarea lui **k** este o valoare nenulă, structura se poate înlocui atât cu structura **cât timp execută**, cât şi cu o structură **repetă - până când**. Evident, la înlocuire, se vor efectua inițializarea, testarea și incrementarea contorului **i**:

```
citeşte n,k   (numere naturale nenule)
n1 ←0 ; n2 ←0 ; p ←1
┌cât timp n ≠ 0 execută
│ ┌dacă n mod 2=0 atunci
│ │ n2 ← n2 + n mod 10*p
│ │altfel
│ │ n1 ← n1 + n mod 10*p
│ └■
│ n ← [n/10]
│ p ← p*10
└■
p ← 1; i←1
┌cât timp i<=k execută
│  p ← p*10
│  i ← i+1
└■
x ← [n2/p] + n1 mod p
scrie x
```

Pentru alegerea unei alte structuri repetitive se acordă 1p., pentru testarea condiției de oprire se acordă 1p., pentru inițializarea contorului se acordă 1p., pentru incrementarea contorului se acordă 1p., iar pentru scrierea integrală a algoritmului se mai acordă 2p. - în total **6p**.

d) Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc.

Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului şi de decizie se acordă câte 1p., pentru instrucţiunea repetitivă se acordă 3p., iar pentru cele 6 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.

2. Un răspuns corect:

s.A.x=s.B.x (Pascal) sau **s.A.x==s.B.x (C/C++)**

Pentru orice răspuns corect se acordă **6p**.

3. Un răspuns corect:

Limbajul Pascal	Limbajul C/C++
`if s<t then writeln(s) else writeln(t)`	`if (strcmp(s,t)<0) cout<<s; else cout<<t;`

Subiectul 3

1. Prelucrarea oferită de subprogram construiește tabloul pe parcursul procesului de descompunere în factori primi. S-au completat componentele vectorului v de la poziția 1, din considerente didactice.

Limbajul Pascal

```
type vector=array[1..100] of integer;
procedure factori(n:longint; var v:vector; var
nf:byte);
var f,e:integer;
begin
   f:=2; nf:=0;
   while n>1 do begin
      e:=0;
      while n mod f=0 do begin
         e:=e+1;
         n:=n div f
      end;
      if e>0 then begin
         inc(nf);
         v[2*nf-1]:=f;
         v[2*nf]:=e
      end;
      inc(f)
   end
end;
```

Limbajul C/C++

```
void factori(int n, int v[], int&nf)
{
   int f=2, e;
   while(n>1) {
      e=0;
      while(n%f==0) {
         e++;
         n=n/f;
      }
      if(e>0) {
         nf++;
         v[2*nf-1]=f;
         v[2*nf]=e;
      }
      f++;
   }
}
```

Pentru respectarea structurii antetului (`procedure`/`void`) se acordă 1p., pentru declararea în ordine a parametrilor `n`, `v` și `nf` și declararea corectă a parametrului de intrare `n` - 1p., pentru declararea corectă a parametrului de ieșire `v` - 1p., pentru declararea corectă a parametrului de ieșire `nf` - 1p., pentru declararea și

	iniţializarea variabilelor locale - 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p. Corectitudinea algoritmică a prelucrării în vederea obţinerii valorilor cerute este notată cu 4p. (factori primi - 2p., exponent - 1p., poziționarea în vector - 1p.). În total **10p.**
2.	Valorile se pot completa pe coloane, conform enunțului. O soluție posibilă, ce va fi realizată ca program în limbajul de programare studiat, care completează liniile și coloanele de la indicele 1, este:

```
pentru i←1,n execută
      pentru j ←1,n execută
            dacă j mod 2 =1 atunci aij←j altfel aij←n+1-j
      sfârșit pentru
sfârșit pentru
```

Pentru declararea corectă a tabloului se acordă 1p., pentru citirea lui **n** se acordă 1p., pentru accesul la un element din tablou se acordă 1p., pentru completarea corectă a unei coloane (pară sau impară) se acordă 1p., pentru completarea corectă a tuturor coloanelor se acordă 2p., iar pentru afișarea matricei conform cerinței se acordă 2p. În plus, pentru declararea variabilelor simple, structura şi corectitudinea sintactică a programului se mai acordă 1p. - în total **10p**.

3. **a)**

Limbajul Pascal

```
var   f:text;
      fr:array[0..9] of longint;
      i,j,x:longint;
      secv:boolean;
begin
      assign(f,'bac.in');reset(f);
      secv:=false;
      while not seekeof(f) do
      begin
            read(f,x);
            if x<10 then begin
                  if not secv then begin
                        secv:=true;
                        for i:=0 to 9 do fr[i]:=0
                  end
            end
            else
                  secv:=false;
            if secv then inc(fr[x])
      end;
      close(f);
      for i:=0 to 9 do
            for j:=1 to fr[i] do
                  write(i,'')
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int x,i,j,secv,fr[10];
int main()
{
      secv=0;
      while(f>>x)
      {
            if(x<10)
            {
                  if(!secv)
                  {
                        secv=1;
                        for(i=0;i<10;i++) fr[i]=0;
                  }
            }
            else
                  secv=0;
            if(secv) fr[x]++;
      }
      f.close();
      for(i=0;i<10;i++)
            for(j=1;j<=fr[i];j++)
                  cout<<i<<' ';
      return 0;
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fișier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea ultimei secvențe, 1p. pentru afișarea corectă a rezultatului și 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare (O(**n**)) și 1p. pentru utilizarea eficientă a memoriei. În total **8p.**

b) O soluție posibilă:

Pe măsură ce citim numerele, contorizăm aparițiile numerelor de o cifră (folosind un vector de frecvențe). La apariția primului număr care are mai mult de o cifră numărătoarea se încheie folosind un semnalizator `secv`, iar la apariția unui nou număr de o cifră, numărătoarea se resetează. În final, vectorul de frecvențe ne permite să afișăm fiecare cifră din ultima secvență, în ordine crescătoare.

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul de numere. Deoarece numărul de valori din fișier (1000000) este dimensiunea care contează în problemă, spunem că am obținut un algoritm liniar (numărul de operații este aproximativ 10∗n). Ca spațiu de memorie, soluția propusă este eficientă, deoarece va utiliza doar un vector de 10 componente, un semnalizator și două-trei variabile simple de lucru.

Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. - în total **2p.**

Menționăm că descrierea coerentă a problemei nu presupune să povestim programul, ci să redăm în limbaj natural ideea de prelucrare.

VARIANTA 15

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Stabiliţi care dintre următoarele expresii Pascal/C/C++ are valoarea true/1 dacă şi numai dacă numărul întreg memorat în variabila z, nu aparţine intervalului (-20,20)U[35,150]. **(4p.)**

	Limbajul Pascal	Limbajul C/C++
a	`(z<-20)OR((z<=34)OR(z>=20))OR(z>150)`	`(z<-20)||((z<=34)||(z>=20))||(z>150)`
b	`(z<-20)OR((z<=35)AND(z>=20))OR(z>=150)`	`(z<-20)||((z<=35)&&(z>=20))||(z>=150)`
c	`(z<-20)OR((z<34)AND(z>20))OR(z>150)`	`(z<-20)||((z<34)&&(z>20))||(z>150)`
d	`(z<=-20)OR((z<35)AND(z>=20))OR(z>150)`	`(z<=-20)||((z<=34)&&(z>=20))||(z>150)`

2. Se consideră tipul de date **elev**, dat prin declararea următoare. Scrieţi o secvenţă de instrucţiuni în urma executării căreia câmpul **calificativ** al variabilei **x** să memoreze unul din şirurile de caractere: **FOARTE BINE**, dacă media aritmetică a câmpurilor **nota1** şi **nota2** este mai mare sau egală cu 9, **BINE**, dacă media aritmetică a câmpurilor **nota1** şi **nota2** este strict mai mică decât 9 şi mai mare sau egală cu 8, respectiv **SUFICIENT**, dacă media aritmetică a câmpurilor **nota1** şi **nota2** este mai mică decât 8. **(6p.)**

Limbajul Pascal

```
type elev = record
  nota1,nota2:integer;
calific:string[12]
end;
var x:elev;
```

Limbajul C/C++

```
struct elev
      {
       int nota1,nota2;
       char calific[13];
       };
elev x;
```

Pentru acces corect la câmpurile înregistrării se acordă **3p.** (pentru fiecare câmp câte 1p), pentru verificare condiţiilor logice corespunzătoare **1p.**, iar pentru memorare şirului de caractere conform cerinţei **2p**. Pentru rezolvare corectă se acordă în total **6p**.

3. Considerăm subprogramul `f` definit mai jos:

Limbajul Pascal

```
function f(x,y:integer):integer;
begin
      if x<=y then f:=x-y
      else f:=f(y-x,x-1)+3*x-2
end;
```

Limbajul C/C++

```
int f(int x, int y)
{
      if (x<=y) return x-y;
      else return f(y-x,x-1)+3*x-2;
}
```

a) Ce valoare are `f(6,12)`? **(3p.)**

b) Dar `f(20,6)`? **(3p.)**

4.	Se consideră un arborele cu rădăcină, cu 11 vârfuri numerotate de la 1 la 11, descris prin următorul vector de "taţi": **(6,5,2,5,0,3,3,3,8,7,7)**. Scrieţi descendenţii direcţi ai rădăcinii şi apoi numărul de frunze ale arborele dat. **(6p.)**
5.	Câte dintre vârfurile grafului neorientat G, reprezentat prin matrice a de adiacenţă de mai jos, au gradul un număr impar? **(4p.)** `0 1 0 0 1` `1 0 1 1 0` `0 1 0 1 1` `0 1 1 0 1` `1 0 1 1 0`

a. 3	**b.** 5	**c.** 2	**d.** 4

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numerel întregi **x** şi **y**, respectiv cu **[a]** partea întreagă a numărului real **a**.

```
citeşte n (număr natural nenul)
nr1←0 ; nr2←0
┌pentru i←1,n execută
│   Citeste x
│   y←x; z←0; t←0
│ ┌ cât timp y≠0 executa
│ │  ┌dacă y%10%2=1 atunci
│ │  │    z←z*10+ y%10
│ │  │    t←1
│ │  └■
│ │   y←[y/10]
│ └■
│ ┌dacă x=z atunci
│ │    nr1←nr1+1
│ │altfel
│ │  ┌dacă t=0 atunci
│ │  │    nr2←nr2+1
│ │  └■
│ └■
└■
scrie nr1+nr2
```

a)	Scrieţi ce se va afişa dacă se citesc, în această ordine, numerele **6, 171, 22, 18, 862, 52, 353**? **(6p.)**
b)	Scrieţi un şir de numere ce pot fi citite astfel încât valoarea afişată să fie **0**. **(4p.)**
c)	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **pentru... execută** cu o structură repetitivă cu test iniţial. **(6p.)**
d)	Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2.	Utilizând metoda backtracking se generează numerele formate din câte 3 cifre distincte din mulţimea {`2,4,6,8`}. Dacă primele trei numere generate sunt, în această ordine: `246`, `248`, `264`, care este cel de-al patrulea număr generat? **(4p.)**			
	a. 426	**b.** 284	**c.** 268	**d.** 468

3.	Fie variabila **s** care memorează şirul de caractere **BACALAUREAT**. Care dintre următoarele expresii au ca valoare caracterul **T**? **(4p.)**	
	Limbajul Pascal	**Limbajul C/C++**
	`a. s[length(s)-1]` `b. s[length(s)-3]` `c. s[5]` `d. s[length(s)]`	`a. s[strlen(s)-2]` `b. s[strlen(s)-4]` `c. s[4]` `d. s[strlen(s)-1];`

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. a) Scrieţi definiţia completă a subprogramului **p** care primeşte prin intermediul parametrului **n** un număr natural nenul **(n≤50),** iar prin intermediul parametrului **a** un tablou unidimensional cu **n** componente întregi, de maximum cinci cifre fiecare. Subprogramul furnizează prin intermediul parametrului **max** valoarea maximă din tabloul **a**, prin intermediul parametrului **imax** indicele valorii maxime din **a**, iar prin intermediul parametrului **sum** suma elementelor din tabloul **a**. **(6p.)**

b) Scrieţi un program care citeşte de la tastatură o valoare naturală nenulă **n**, **(3≤n≤50),** apoi cele **n** elemente, distincte, ale unui tablou unidimensional **x**. Fiecare dintre aceste elemente este un număr natural având cel mult cinci cifre. Folosind apeluri utile ale subprogramului **p**, programul calculează şi afişează pe ecran media aritmetică a elementelor care ar rămâne în tabloul **x** dacă s-ar elimina valoarea maximă, iar pe poziţia elementului maxim s-ar insera valoarea **1**. Valoarea cerută se va afişa cu 2 zecimale.

Exemplu: dacă se citesc pentru **n** valoarea `5`, iar pentru tabloul **x** valorile **(**`10,8,14,7,5`**),** programul va afişa `6.20` (deoarece `10+8+7+5+1)/5=6.20`). **(4p.)**

2. Scrieţi un program care citeşte de la tastatură un număr natural **n** (1≤**n**≤50), apoi elementele unei matrice cu **n** linii şi **n** coloane, numere întregi din intervalul **[-100,100]** şi afişează pe ecran media aritmetică a elementelor strict pozitive, situate pe linii pare, sub diagonala secundară în matricea dată. Dacă nu există elemente strict pozitive pe linii pare sub diagonala secundară, se va afişa mesajul "**NU EXISTA**".

Exemplu: pentru **n=4** şi matricea de mai jos:

	`1  2  4  5` `0  6  3  2` `2  4 -2  0` `3 -5  4 -3` se afişează valoarea **3** (sunt luate în considerare la calculul mediei doar elementele marcate, **(2+4)/2=3**). **(10p.)**
3.	Se consideră fişierul **BAC.TXT** ce conţine pe prima linie un număr natural **n (10≤n≤1000000)** şi pe a doua linie un şir **descrescător** cu numere naturale de cel mult nouă cifre fiecare, separate prin câte un spaţiu. a) Să se scrie un program care, folosind un algoritm eficient din punct de vedere al memoriei utilizate şi al timpului de executare, citeşte din fişier toţi termenii şirului şi afişează toate perechile distincte de termeni distincţi alăturaţi ai şirului dat care au valori mai mici strict decât **n**. Perechile se vor scrie pe ecran, câte cinci perechi pe o linie, separate prin câte un spaţiu, iar pe ultima linie se vor scrie atâtea perechi câte mai rămân. **(8p.)** **Exemplu:** dacă fişierul **BAC,TXT** are următorul conţinut `334` `2289 22890 22890 356 356 321 321 321 234 234 57 57 57 40 35 32 23 12` Programul va afişa pe ecran: `(321,234) (234,57) (57,40) (40,35) (35,32)` `(32,23) (23,12)` b) Descrieţi în limbaj natural metoda de rezolvare folosită explicând în ce constă eficienţa ei (3-4 rânduri). **(2p.)**

Observaţii:

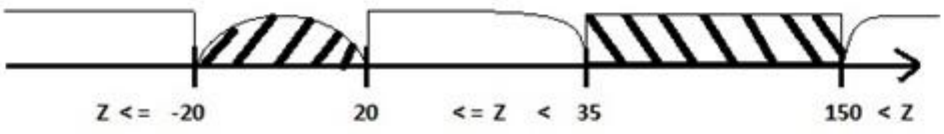
• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 15 - *Rezolvare*

Subiectul 1

1.

Pentru ca variabila z **să aparţină** reuniunii intervalelor (-20,20), [35,150] trebuie ca z să fie mai mare strict ca -20 şi mai mic strict decât 20, sau z să fie mai mare sau egal ca 35 şi mai mic sau egal decât 150, deci să avem relaţia: (z>-20) AND (z<20) OR (z>=35) AND (z<=150) (vezi în desen, cele două intervale haşurate)



Vom nega toată relaţia de mai sus, obţinând astfel o nouă relaţie echivalentă cu prima şi anume: NOT((z>-20) AND (z<20) OR (z>=35) AND (z<=150)). Astfel, pentru ca variabila z **să nu aparţină** reuniunii intervalelor (-20,20), [35,150] trebuie ca z să fie mai mic sau egal ca -20 sau mai mare sau egal decât 20 şi z să fie mai mic strict ca 35 sau mai mare strict decât 150. Această relaţie este echivalentă cu următoarea expresie: (z<=-20) OR (z>=20) AND (z<35) or (z>150) (vezi în desen, cele trei intervale nehaşurate)

Pentru precizarea răspunsului **d)** se acordă **4p**.

2.

Limbajul Pascal

```
if (x.nota1+x.nota2)/2>=9 then
      x.calific:='FOARTE BINE'
else
      if (x.nota1+x.nota2)/2>=8 then
            x.calific:='BINE'
      else
            x.calific:='SUFICIENT'
```

Limbajul C/C++

```
if((x. nota1+x.nota2)/2>=9)
      x.calific="FOARTE BINE"
else
      if((x.nota1+x.nota2)/2>=8)
            x.calific="BINE"
      else
            x.calific="SUFICIENT";
```

3.

a) Pentru răspunsul **-6** se acordă **3p**.

b) Apelurile recursive sunt: `f(20,6) = f(6-20,20-1)+3*20-2 = f(-14,19)+60-2=-14-19+58=-33+58=25`

Pentru răspunsul **25** se acordă **3p.**

4.

Din vectorul de "taţi" dat rezultă că rădăcina arborelui este nodul 5 (deoarece `tata[5]=0`). Parcurgem vectorul din nou şi căutăm nodurile care îl au ca tată pe 5. Acestea sunt nodurile 2 şi 4 (`tata[2]=5` şi `tata[4]=5)`.

Nodurile cuprinse între 1 şi 11 care nu apar în vectorul "tata" sunt frunzele arborelui: 1, 4, 9, 10 şi 11.

Pentru precizarea răspunsului **2 4 5** sau **4 2 5** se acordă **6p (**câte 1p. pentru fiecare dintre descendenţii rădăcinii, 1p. pentru neprezicarea vreunui descendent suplimentar, 2p. pentru numărul de frunze şi 1p. pentru respectarea ordinii valorilor).

5.

Gradul fiecărui nod se poate calcula însumând elementele de pe linia sau coloana corespunzătoare din matricea de adiacenţă.

Pentru răspunsul **d)** se acordă **4p.**

Subiectul 2

1.

a) Variabila **nr1** reţine numărul de numere cu toate cifrele impare şi care sunt palindrom (un număr palindrom este numărul care, citit de la stânga la dreapta, este egal cu numărul citit de la dreapta la stânga), iar variabila **nr2** reţine numărul de numere cu toate cifrele pare. Pentru valorile date, **nr1=2** (**171** şi **353**), iar **nr2=2** (**22** şi **862**). Rezultatul afişat este suma variabilelor **nr1** şi **nr 2**, adică **4**.

Pentru precizarea valorii **4** se acordă **6p**.

b) Metoda de lucru presupune să găsim un şir de **n** numere în care niciun element care este palindrom nu are toate cifrele impare, şi niciun număr nu are toate cifrele pare. Răspunsul poate fi orice şir de numere cu această proprietate.

Pentru precizarea unui astfel de şir, de exemplu, **3 252 34 43**, se acordă **4p**.

c) Înlocuirea structurii `pentru i<-1,n execută` cu o structură cu test iniţial, presupune utilizarea unei structuri repetitive `cât timp... execută`. Astfel trebuie să avem atât grija iniţializării variabilei `i`, a incrementării variabilei `i`, cât şi a testării condiţie logice de ieşire din buclă.

Algoritmul cerut este:

```
citeşte n (număr natural nenul)
nr1←0; nr2←0
i←1
┌cat timp i<=n execută
│    Citeste x
│    y←x; z←0; t←0
│   ┌ cât timp y≠0 executa
│   │ ┌dacă y % 10 % 2 = 1 atunci
│   │ │     z←z*10+ y % 10; t←1
│   │ └■
│   │  y←[y/10]
│   └■
│   ┌dacă x=z atunci
│   │   nr1←nr1+1
│   │altfel
│   │ ┌dacă t=0 atunci
│   │ │    nr2←nr2+1
│   │ └■
│   │ i←i+1;
│   └■
└■
scrie nr1+nr2
```

Pentru alegerea structurii repetitive corecte se acordă 2p., pentru iniţializarea lui i se acordă 1p., pentru testarea condiţiei de continuare sau oprire se acordă 1p., pentru incrementarea lui i se acordă 1p., iar pentru scrierea integrală a restului algoritmului se mai acordă 1p. - în total **6p**.

d) Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deşi scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerinţă se valorifică experienţele anterioare de implementare şi testare a programelor pe calculator, experienţe care au cimentat cunoştinţele privind structura programelor, declararea variabilelor, sintaxa instrucţiunilor programului, regulile de scriere a expresiilor etc.

Pentru instrucţiunile corecte de citire a datelor, de afişare a rezultatului se acordă câte 1p., pentru fiecare structură de decizie se acordă câte 1p., pentru fiecare structură

		repetitivă se acordă câte 2p., iar pentru scrierea corectă a tuturor atribuirilor se acordă 1p. Pentru de declarare a variabilelor şi structura corectă a programului se mai acordă 1p., în total **10p**.
2.	Această problemă este o aplicaţie la problema aranjamentelor de 4 cifre luate câte 3. Se generează în maniera "înaintare cu revenire", numerele: 246, 248, 264, **268**, etc. Pentru răspunsul **c. 268** se acordă **4p**	
3.	Cu ajutorul funcţiei **length(s)/strlen(s)** vom depista lungimea şirului de caractere **s**, respectiv poziţia ultimului caracter (adică **11**). Cum la un şir de caractere ne putem referi ca la un vector, atunci expresia **s[length(s)]** (limbajul Pascal) / **s[strlen(s)-1]** (limbajul C++) returnează ultimul caracter din şirul s, adică **T**. Pentru răspunsul **d)** se acordă **4p.**	

Subiectul 3

1. Subprogramul **p** prelucrează elementele vectorului **a** cu **n** elemente determinând valoarea maximă în variabila **max**, indicele de unde provine aceasta în variabila **imax**, precum şi suma elementelor vectorului în variabila **s** (iniţial presupunem ca fiind maximă prima valoare din vector, apoi comparăm vechiul maxim cu următoarea valoare din vector **`a[i]`** şi dacă aceasta este mai mare atunci maximul se schimbă şi devine **`a[i]`**.

Când schimbăm maximul, schimăm şi indicele de unde provine acesta). La o singură parcurgere a vectorului vom adăuga şi la suma **s** câte un element al vectorului, suma iniţial având prima valoare din vector.

Limbajul Pascal

```
type vector=array[1..50] of longint;
var   max,s:longint;
      x:vector;
      n,imax,i:byte;
procedure p(n:byte; a:vector; var max:longint; var imax:byte; var sum:longint);
var i:byte;
begin
      max:=a[1]; imax:=1; sum:=a[1];
      for i:=2 to n do begin
            if max<a[i] then
            begin
                  max:=a[i];
                  imax:=i;
            end;
            sum:=sum+a[i];
      end
end;
begin
      readln(n);
      for i:=1 to n do
            read(x[i]);
      p(n,x,max,imax,s);
      s:=s-max+1;
      writeln('Media aritmetica=',s/n:4:2);
end.
```

	Limbajul C/C++ ``` long max,x[50]; double s; int n,imax,i; void p(int n, int a[50], long &max, long &imax, double &sum) { int i; max=a[1];imax=1; sum=a[1]; for(i=2;i<=n;i++) { if (max<a[i]) { max=a[i]; imax=i; } sum=sum+a[i]; } } void main() { cin>>n; for(i=1;i<=n;i++) cin>>x[i]; p(n,x,max,imax,s); s=s-max+1; cout<<" Media aritmetica="<<s/n; } ``` Pentru respectarea structurii antetului (**procedure**/**void**) se acordă **1p**., pentru declararea corectă a parametrilor de intrare **n** şi **a** încă **1p**., pentru declararea corectă a parametrilor de ieşire **max, imax, sum** încă **1p**., pentru declararea variabilelor locale încă **1p.**, iar pentru respectarea structurii subprogramului şi a sintaxei limbajului se acordă încă **1p**. Corectitudinea algoritmică a prelucrării în vederea obţinerii valorii cerute este notată cu **4p**. Pentru transmiterea rezultatelor prin parametrii **max, imax, sum** se acordă **1p**. În total **10p.**
2.	O metodă uzuală vizitează toate elementele matricei, considerând doar pe acelea care apar pe linii pare şi adăugăm la sumă doar elementele pozitive situate sub diagonala secundară: ``` s←0; nr←0 pentru i←1,n execută daca i %2=0 atunci pentru j←1,n execută daca (i+j>n+1) si (a[i,j]>0) atunci s←s+a[i,j] nr←nr+1 sfârșit dacă sfârșit pentru sfârșit dacă sfârșit pentru daca s=0 atunci Scrie ' NU EXISTA' altfel Scrie s/nr sfârșit dacă ```

O altă metodă vizitează doar elementele matricei situate sub diagonala secundară şi consideră doar liniile pare. Aceasta este echivalentă cu metoda anterioară, dar este mai eficientă.

```
s←0; nr←0
pentru i←2,n,2 execută
      pentru j←n+1-i,n execută
            dacă a[i,j]>0 atunci
                  s←s+a[i,j]
                  nr←nr+1
            sfârșit dacă
      sfârșit pentru
sfârșit pentru
daca s=0 atunci
    Scrie ' NU EXISTA'
altfel
    Scrie s/nr
sfârșit dacă
```

Structura repetiitvă cu contorul `i` având pasul 2 se poate implementa în Pascal astfel:

```
for k:=1 to n div 2 do begin
      i:=2*k
      ...
end
```

Pentru declararea corectă a matricei se acordă **1p**., pentru citirea lui n şi citirea matricii cu **n*n** elemente se mai acordă **2p**., pentru parcurgerea corectă a matricei se acordă **1p**., pentru calculul corect al mediei aritmetice **1p.**, pentru scrierea corectă a condiţiilor de elemente pozitive, situate sub diagonala secundară şi pe linii pare **1p**., accesare corectă a unui al tabloului - **1p**, iar pentru afişarea matricei conform cerinţei se acordă **2p**. În plus, pentru declararea variabilelor simple, structura şi corectitudinea sintactică a programului se mai acordă **1p**. - în total **10p**.

3. **b)**

Vom evita utilizarea vectorilor, considerând doar variabile simple. Vom parcurge fişierul şi, pentru fiecare valoare **x** citită care este diferită de următoarea valoare citită **y**, vom afişa perechi de forma **(x,y)**, numărând totodată aceste perechi. Când numărul perechilor este divizibil cu **5**, vom trece la linie nouă.

Eficienţa algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată şirul de numere (algoritm liniar). Ca spaţiu de memorie, soluţia propusă este eficientă, deoarece va utiliza doar variabile simple de lucru, în locul unui vector, evitând astfel memorarea celor maximum 1000000 de numere din fişier.

Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienţei, încă 1p. - în total **2p.**

a) **Limbajul Pascal**

(vezi programul pe pagina următoare)

```
var f:text;
      nr,x,y,n:longint;
begin
      assign(f,'BAC.TXT');reset(f);
      readln(f,n);
      read(f,x);
      nr:=0;
      while not eof(f) do begin
            read(f,y);
            if (x<>y) and (x<n) and (y<n) then begin
                  write('(',x,',',y,') ');
                  nr:=nr+1;
                  if nr mod 5=0 then writeln;
            end;
            x:=y;
      end;
      close(f);
end.
```

Limbajul C/C++

```
#include<fstream.h>
void main()
{
      long nr=0,x,y,n;
      ifstream f("bac.txt");
      f>>n;
      f>>x;
      while (f>>y)
      {
            if ((x!=y) and (x<n) and (y<n))
            {
                  cout<<"("<<x<<","<<y<<")";
                  nr++;
                  if (nr%5==0) cout<<endl;
            }
            x=y;
      }
      f.close();
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect şi deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea tuturor perechilor cerute, 1p. pentru afişarea lor în formatul cerut şi 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucţiunilor, etc.).

Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare ($O(\mathbf{n})$) şi 1p. pentru utilizarea eficentă a memoriei. În total **8p.**

Menţionăm că prin **algoritm principial corect** se înţelege un algoritm care, în intenţie, urmăreşte să rezolve corect problema, deşi poate să prezinte scăpări logice sau confuzii.

VARIANTA 16

Subiectul 1 (20p)

<table>
<tr><td colspan="5">Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte</td></tr>
<tr><td>1.</td><td colspan="4">Care este valoarea expresiei Pascal/C/C++ de mai jos? (4p.)</td></tr>
<tr><td></td><td colspan="2">Limbajul Pascal
93 div 10 * 3 +789 mod 10 / 2</td><td colspan="2">Limbajul C/C++
93 / 10 * 3 +789 % 10 / 2.0</td></tr>
<tr><td>a.</td><td colspan="2">32.25</td><td colspan="2">32.25</td></tr>
<tr><td>b.</td><td colspan="2">23</td><td colspan="2">23</td></tr>
<tr><td>c.</td><td colspan="2">31.50</td><td colspan="2">31.50</td></tr>
<tr><td>d.</td><td colspan="2">31</td><td colspan="2">31</td></tr>
<tr><td>2.</td><td colspan="4">Funcţia f are definiţia alăturată. Ce valoare are f(17)? (4p.)</td></tr>
<tr><td></td><td colspan="2">Limbajul Pascal</td><td colspan="2">Limbajul C/C++</td></tr>
<tr><td></td><td colspan="2">function f(x:integer):longint;
begin
if x<=9 then f:=0
else
    if x mod 4=0 then f:=1
    else f:=f(x-3)*30+4;
end;</td><td colspan="2">long f(int x)
{
    if(x<=9) return 0;
    else
      if(x%4==0) return 1;
      else return f(x-3)*30+4;
}</td></tr>
<tr><td></td><td>a. 34</td><td>b. 3724</td><td>c. 6543</td><td>d. 165</td></tr>
<tr><td>3.</td><td colspan="4">Profesorul doreşte să aşeze cei cinci elevi ai unei grupe, Alina, Bogdan, Beatrice, Andrei şi Mihai, în „şir indian”. Utilizând metoda backtracking, se generează toate posibilitățile de a forma astfel de șiruri astfel încât în fiecare aşezare Beatrice să fie în fața lui Andrei. Primele două soluţii generate sunt, în această ordine: (Alina, Bogdan, Beatrice, Andrei, Mihai), (Alina, Bogdan, Beatrice, Mihai, Andrei). Următoarea soluţie generată este: (4p)</td></tr>
<tr><td></td><td colspan="2">a. (Alina, Bogdan, Andrei, Beatrice, Mihai)</td><td colspan="2">b. (Alina, Bogdan, Mihai, Beatrice, Andrei)</td></tr>
<tr><td></td><td colspan="2">c. (Alina, Bogdan, Mihai, Beatrice, Andrei)</td><td colspan="2">d. (Alina, Andrei, Bogdan, Beatrice, Mihai)</td></tr>
<tr><td>4.</td><td colspan="4">Se consideră arborele dat prin următorul vector ”de taţi”: (7, 7, 0, 3, 7, 5, 3, 6). Ştiind că rădăcina se află pe nivelul 0, stabiliţi care sunt frunzele aflate pe nivelul 2: (4p)</td></tr>
<tr><td></td><td>a. 1 2 5</td><td>b. 1 2</td><td>c. 2 5</td><td>d. 4 7</td></tr>
</table>

5.	Câte dintre vârfurile grafului neorientat G, reprezentat prin matrice a de adiacenţă de mai jos, au gradul un număr impar? **(4p)** `0 1 0 0 1` `1 0 1 1 0` `0 1 0 1 1` `0 1 1 0 1` `1 0 1 1 0`			
	a. 3	**b.** 5	**c.** 2	**d.** 4

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numerelor întregi **x** şi **y.**

```
citeşte x,y (numere naturale nenule, x<y)
s←0
┌pentru i←x,y execută
│   ┌dacă i % 5 =0 atunci
│   │    s←s+i*(i+1)
│   │     scrie s
│   └■
└■
```

a)	Ce şir de valori se va afişa dacă se citesc valorile **2** şi **12**? **(6p.)**
b)	Pentru **x=30**, scrieţi toate valorile care pot fi introduse pentru **y** astfel încât rezultatul afişat să fie **930 2190** **(6p.)**
c)	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **pentru ......execută** cu o structură repetitivă de alt tip. **(6p.)**
d)	Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. Se consideră declarările de mai jos, în care variabila **e** memorează date despre un anumit elev. Scrieți instrucțunea prin care se iniţializează ziua naşterii acestui elev cu valoarea 19. **(6p.)**

Limbajul Pascal	Limbajul C/C++
`type data=record` `      zi:integer;` `      luna:integer;` `      an:integer` `end;` `type elev=record` `   nume:string[30];` `   data_nasterii:data;` `   media:real` `end;` `var e:elev;`	`struct data{` `      int zi;` `      int luna;` `      int an;` `};` `struct elev{` `   char nume[30];` `   data data_nasterii;` `   float media;` `}e;`

3. Ştiind că variabila **k** este de tip întreg şi că variabila **s** reţine şirul de caractere **abracadabra**, scrieţi ce se va afişa la executarea secvenţei alăturate. **(6p.)**

	Limbajul Pascal	Limbajul C/C++
	`for k:=1 to length(s) do` `if (s[k]='r') then` `s[k]:='b';` `write(s);`	`for(k=0;k<=strlen(s);k++)` `if (s[k]=='r')` `s[k]='b';` `cout<<s; | printf("%s",s);`

Subiectul 3 (30p)

	Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.
1.	Se consideră subprogramul **sub**, cu doi parametri: **– n**, prin care primește un număr natural cu cel mult nouă cifre (n>1); **– c**, prin care furnizează: ultima cifră a lui n, dacă n este prim sau prima cifră a lui n, în caz contrar; Scrieţi definiţia completă a subprogramului. Exemplu:pentru n=17, după apel c=7, iar pentru n=12, după apel c=1. **(10p.)**
2.	Scrieţi un program Pascal care citeşte de la tastatură un număr natural **n (2<n<=10)** şi construieşte în memorie o matrice **A** cu **n** linii şi **n** coloane în care toate elementele de pe prima linie şi prima coloană au valoarea **2** şi oricare alt element A_{ij} din matrice este egal cu suma a 2 elemente: primul situat pe linia **i-1** si coloana **j**, iar al doilea situat pe linia **i** si coloana **j-1**, ca în exemplu. Matricea va fi afişată pe ecran, linie cu linie, numerele de pe aceeaşi linie fiind separate prin câte un spaţiu. **(10p.)** **Exemplu:** pentru **n=4**, se afişează matricea: 2 2 2 2 2 4 6 8 2 6 12 20 2 8 20 40
3.	Se citeşte din fişierul **bac.txt** un număr natural **n (n≤**30000**)** aflat pe prima linie a fişierului şi apoi **n** cifre separate prin câte un spaţiu, aflate pe a doua linie a fişierului. Se cere să se afişeze pe ecran cifrele care apar în fișier, în ordine descrescătoare, fiecare cifră fiind urmată de atâtea caractere '*****' câte apariții are cifra respectivă în șirul dat. Exemplu: dacă fişierul `bac.txt` are conţinutul: `18` `2 3 3 0 9 1 2 1 3 7 1 5 2 7 1 0 3 2` se va afişa pe ecran `9*7**5*3****2****1****0**` a) Descrieţi pe scurt un algoritm de rezolvare al problemei, eficient din punct de vedere al spaţiului de memorie utilizat şi al timpului de executare, explicând în ce constă eficienţa metodei alese. **(2p.)** b) Scrieţi programul Pascal/C/C++ corespunzător algoritmului descris. **(8p.)**

VARIANTA 16 - *Rezolvare*

Subiectul 1

1. Se evaluează expresia de la stânga la dreapta ținându-se cont de prioritatea operatorilor, adică mai întâi se efectuează calculul pentru cei din grupa operatorilor multiplicativi (mod,div,/) în ordinea în care se întâlnesc în expresie de la stânga la dreapta, și la o nouă parcurgere a expresiei de la stânga la dreapta, se efectuează calculul pentru cei din grupa operatorilor aditivi (+,-). Astfel, la finalul evaluărilor, se obține un număr real. Adunarea unui număr întreg cu un număr real are ca rezultat un număr real.

Pentru precizarea răspunsului **c)** se acordă **4p**.

2. Se urmăresc apelurile recursive:

f(17)= f(14)*30+4 ; f(14)=f(11)*30+4 ; f(11)=f(8)*30+4 ; f(8)=0

La revenire se fac calculele:

f(11)= 0*30+4=4 ; f(14) =4*30+4=124; f(17)=124*30+4 = 3720+4 = 3724

Pentru răspunsul **b**.`3724` se acordă **4p**.

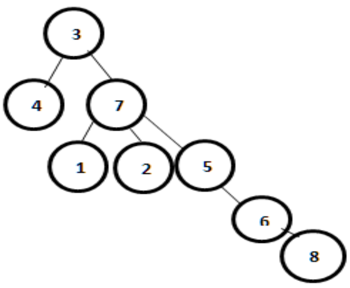
3. Se generează permutările de câte 5 elemente în maniera ”înaintare cu revenire”, reținând în vector indicii vectorului de șiruri de caractere, adică numere {1,2,3,4,5}.Astfel, soluțiile generate, fără impunerea condiției ca Beatrice să fie în fața lui Andrei(indicii 3 și 4 ai șirului dat) sunt:

1. (Alina, Bogdan, Beatrice, Andrei, Mihai), adica indicii: 1, 2, 3, 4, 5
2. (Alina, Bogdan, Beatrice, Mihai, Andrei) , adica indicii: 1, 2, 3, 5, 4
3. ~~(Alina, Bogdan, Andrei, Beatrice, Mihai)~~ , adica indicii: ~~1, 2, 4, 3, 5~~
4. ~~(Alina, Bogdan, Andrei, Mihai, Beatrice)~~ , adica indicii: ~~1, 2, 4, 5, 3~~
5. (Alina, Bogdan, Mihai, Beatrice, Andrei) , adica indicii: 1, 2, 5, 3, 4

Dar indicele 3 trebuie să fie în fața indicelui 4 în soluțiile obținute, deci soluțiile 3. și 4. nu sunt valide, le tăiem.

Pentru răspunsul **b**. (Alina, Bogdan, Mihai, Beatrice, Andrei) se acordă **4p**.

4. Din vectorul de referințe ascendente deducem că rădăcina arborelui este nodul 3 (**`tata[3]=0`**). La o nouă parcurgere căutăm nodurile care îl au ca tată pe 3. Aceste sunt nodurile 4 și 7 (**`tata[4]=3`** și **`tata[7]=3)`**. Deci, deoarece nodul 3 este nodul de pe nivelul 0, atunci nodurile 4 și 7 se află pe nivelul 1. La o nouă investigare a vectorului, aflăm nodurile care îl au ca părinte pe oricare dintre nodurile 4 și 7. Doar nodul 7 este părinte pentru nodurile 1, 2 si 5 de pe nivelul 2, iar nodul 4 nu este parintele niciunui nod. Deci există trei noduri aflate pe nivelul 2 dintre care doar două sunt frunze, și anume 1 și 2 (nodul 5 nu este frunza, deși se află pe nivelul 2)



Pentru precizarea răspunsului b**)** se acordă **4p**.

5.	Gradul fiecărui nod se poate calcula însumând elementele de pe linia sau coloana corespunzătoare din matricea de adiacenţă.Pentru răspunsul **d)** se acordă **4p**

Subiectul 2

<table>
<tr><td rowspan="4">1.</td><td>a)</td><td>Pentru valorile date, x=2 și y=12, algoritmul alege, dintre numerele cuprinse între 2 și 12, pe cele divizibile cu 5. Rezultă s=5*6=30 și apoi s=30+10*11=140.

Pentru precizarea valorilor 30 140 se acordă 6p.</td></tr>
<tr><td>b)</td><td>Algoritmul determină suma produselor a doi termeni consecutivi ai numelelor care sunt multipli si lui 5, numere cuprinse între x și y - citite.

Pentru x=30, valorile posibile pentru y pot fi 30,31, 32, 33 etc.

Pentru i=30, se va afișa 930. Ne punem întrebarea care este valoarea lui y pentru care se va afișa și valoarea 2190? Pentru i=35 se va calcula si apoi afișa s=s+35*36=930+1260 =2190, Obsevăm că i poate lua valorile 35,36,37,38,39 și rezultatul afișat rămâne cel corect. Dacă i ar primi și valoarea 40, rezultatul afișat ar conține trei valori (930 2190 și 3830).

Pentru precizarea valorilor 35, 36, 37, 38, 39 se acordă 4p. Pentru cel puțin o valoare corectă sau pentru rezultatul 5 (în locul celor 5 valori) se acordă 2p., Pentru 2-4 valori corecte se acordă 3p.</td></tr>
<tr><td>c)</td><td>Înlocuirea structurii pentru .... execută cu o structură cu test inițial, presupune inițializarea, testarea și incrementarea explicită a variabilei i:

<pre>
citeşte x,y (numere naturale nenule, x<y)
s←0
i←x
┌cat timp i<=y execută
│   ┌dacă i % 5 =0 atunci
│   │   s←s+i*(i+1)
│   │   scrie s
│   └■
│   i←i+1
└■
</pre>
Pentru alegerea unei structuri repetitive (cât timp/repeta) se acordă 1p., pentru inițializarea lui i se acordă 1p., pentru testarea condiției de continuare sau oprire se acordă 1p., pentru incrementarea lui i se acordă 1p., pentru corectitudine globală a algoritmului se acordă 1p., iar pentru scrierea integrală a restului algoritmului se mai acordă 1p. – în total 6p.</td></tr>
<tr><td>d)</td><td>Se testează capacitatea de a reprezenta pe hârtie algoritmul dat, cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, experiențe care au cimentat cunoștințele privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor etc.

Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și atribuiri se acordă câte 1p., pentru instrucţiunea repetitivă se acordă 3p., iar pentru instrucțiunea de decizie se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total 10p.</td></tr>
<tr><td>2.</td><td colspan="2">Pentru ca referirea la câmpul an să fie corectă trebuie să începem cu variabila declarată, adică e (nu cu numele tipului – elev- și nici cu vreun alt câmp).

Dar variabila e este de tipul de date elev, iar elev are 3 câmpuri: nume, media,</td></tr>
</table>

	data_nasterii – de tip **data**. Dar tipul **data** are 3 câmpuri: **zi**, luna, an. Deci referirea corectă la câmpul cerut este: **e.data_nasterii.zi,** iar atribuirea corectă este e.data_nasterii.zi:=19; sau e.data_nasterii.zi=19; Pentru accesul corect la câmpuri se acordă 4p., pentru atribuirea corectă se acordă încă 2p., în total **6p.**
3.	Algoritmul dat înlocuiește fiecare caracter din șirul **s** care este **'r'** cu caracterul **'b'.** La finalul executării, șirul s, care inițial avea valoarea **'abracadabra',** va conține **'abbacadabba'** Pentru răspuns corect și anume **'abbacadabba'** se acordă **6p**. Se acordă doar **3p** pentru înlocuirea corectă a unui singur caracter **'r'**.

Subiectul 3

1\. Prelucrarea oferită de subprogram testează proprietatea de număr prim a lui **n** și determină, după caz, prima sau ultima cifră a lui n. Valoare care este returnată de subprogram prin intermediul parametrul **c**.

Limbajul Pascal

```
procedure sub(n:longint; var c:byte);
var i:integer; prim:boolean;
begin
      prim:=true;
      for i:=2 to trunc(sqrt(n))do
            if n mod i = 0 then
            begin
                  prim:=false;
                  break;
            end;
      if prim then
            c:=n mod 10
      else begin
            while n>=10 do
                  n:=n div 10;
            c:=n;
      end;
end;
```

Limbajul C/C++

```
void sub(long n, int &c)
{
      int i,prim=1;
      for(i=2;i*i<=n;i++)
            if (n % i == 0)
            {     prim=0;
                  break;
            }
      if (prim==1)
            c=n % 10
      else
      {     while (n>=10)
                  n=n/10;
            c=n;
      }
}
```

Pentru respectarea structurii antetului (`procedure`/`void`) se acordă 1p., pentru declararea corectă a parametrului de intrare **n** se acordă 1p., pentru declararea corectă a parametrului

de ieșire **c** încă 1p., pentru declararea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p. Corectitudinea algoritmică a prelucrării în vederea obținerii valorii cerute este notată cu 4p. Pentru transmiterea naturală a rezultatului prin parametrul **c** se acordă 1p. În total **10p.**

2.

Se completează inițial linia 1 și coloana 1 cu valoarea 2, elemente de care depind calcularea corectă a celorlalte elemente situate pe liniile 2,3...n, respectiv coloanele 2,3...n.

Apoi se completează și restul elementelor din matrice, folosindu-se formula descrisă în enunț: `a[i,j]:= a[i-1,j]+ a[i,j-1];`

Programul, în maniera descrisă, este:

Limbajul Pascal

```
var   a:array[1..10,1..10] of integer;
      i,j,n:integer;
begin
      readln(n);
      for j:=1 to n do begin
            a[j,1]:=2;
            a[1,j]:=2;
      end;
      for i:=2 to n do
            for j:=2 to n do
                  a[i,j]:= a[i-1,j]+ a[i,j-1];
      for i:=1 to n do begin
            for j:=1 to n do
                  write(a[i,j],’ ‘);
            writeln;
      end;
end.
```

Limbajul C/C++

```
#include <iostream.h>
int a[11][11],i,j,n;
void main()
{
      cin>>n;
      for(j=1;j<=n;j++)
      {
            a[j][1]=2;
            a[1][j]=2;
      }
      for(i=2;i<=n;i++)
            for(j=2;j<=n;j++)
                  a[i][j]:= a[i-1][j]+ a[i][j-1];
      for(i=1;i<=n;i++)
      {
            for(j=1;j<=n;j++)
                  cout<<a[i][j]<<' ';
            cout<<endl;
      }
}
```

Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui n se mai acordă 1p., pentru parcurgerea unei coloane sau linii se acordă 1p., pentru completarea corectă a unei coloane sau linii se acordă 2p., pentru completarea corectă a matricei se acordă 2p., iar pentru afișarea matricei conform cerinței se acordă 2p. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă 1p. – în total **10p**.

<table>
<tr><td>3.</td><td>a)</td><td>
Vom utiliza un vector ap cu ajutorul căruia vom număra aparițiile cifrelor citite de pe a doua linie din fișier, cu indicii între 0 și 9, vector care are inițial toate elementele egale cu 0.

Vom parcurge fișierul și, pentru fiecare valoare întreagă x citită, vom incrementa în vectorul ap pozitia x, valoarea ap[x] memorând la finalul citirii numărul total de apariții ale elementului x în șirul de cifre dat. La final, vom parcurge vectorul ap descrescător și vom afișa doar elementele x a căror apariție este mai mare decât 0 și, după fiecare, numărul de steluțe dat de valoarea ap[x].

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul de numere. Deoarece numărul de valori din fișier (n) este dimensiunea care contează în problemă, spunem că am obținut un algoritm liniar în raport cu n. Ca spațiu de memorie, soluția propusă este eficientă, deoarece va utiliza, pe lângă variabile simple de lucru, doar un vector cu 10 de componente, evitând astfel memorarea celor maximum 30000 de numere din fișier.

Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. – în total 2p.

Menționăm că descrierea coerentă a problemei nu presupune să povestim programul, ci ideea de prelucrare. În acest sens e mai bine să explicăm că “parcurgem”, “numărăm”, ”căutăm primul element care …” etc. decât să dăm detalii tehnice cum ar fi ”folosim un for”, ”incrementăm variabila întreagă i…” etc.
</td></tr>
<tr><td></td><td>b)</td><td>
Limbajul Pascal
<pre>
var   f:text; c:byte;
      ap:array[0..9]of integer;
      n,i,j:integer;
begin
      assign(f,'bac.txt');reset(f);
      readln(f,n);
      for i:=0 to 9 do ap[i]:=0;
      for i:=1 to n do
      begin
            read(f,x);
            ap[x]:= ap[x]+1;
      end;
      close(f);
      for i:=9 downto 0 do
            if ap[i]>=1 then begin
                  write(i);
                  for j:=1 to ap[i] do write(‘*’);
            end;
end.
</pre>
Limbajul C/C++
<pre>
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.txt");
int ap[10],n,i,j;
int main()
{   for(i=0;i<10;i++) ap[i]=0;
      f>>n;
      for(i=1;i<=n;i++)
      {
            f>>x;
            ap[x]=ap[x]+1;
      }
      f.close();
</pre>
</td></tr>
</table>

		`for(i=0;i<10;i++)` `    if(ap[i]>=1){` `        cout<<i;` `        for(j=1;j<=ap[i];j++)` `            cout<<'*';` `    }` `}` Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea tuturor valorilor cerute, 1p. pentru afișarea numerelor si a caracterelor '*' în ordine strict descrescătoare și 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor etc.). Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare (O(**n**)) și 1p. pentru utilizarea eficentă a memoriei. În total **8p.** Menționăm că prin algoritm principial corect se înțelege un algoritm care, în intenție, urmărește să rezolve corect problema, deși poate să prezinte scăpări logice sau confuzii.

VARIANTA 17

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Indicaţi expresia care are valoarea **true/1** dacă şi numai dacă numărul real memorat în variabila **x** se află în intervalul [-5,5].

Variabila **x** este de tip întreg și poate memora un număr natural cu cel mult două cifre. Stabiliți valoarea maximă pe care o poate avea expresia de mai jos.

Limbajul Pascal	Limbajul C/C++
`x mod 3 + (x +1) mod 3+(x+2) mod 3`	`x%3 +(x+1)%3+(x+2)%3`

a. 1	b. 2	c. 3	d. 99

2. Pentru subprogramul **f** cu definiția următoare, indicați conținutul variabilei x după apelul **f(1078, x);**

Limbajul Pascal

```
procedure f( n:longint;var x:longint);
begin
  if n>0 then
     if n mod 2=1 then
        begin
          x:=x*10+n mod 10;
          f(n div 10,x);
         end
       else
         begin
           f(n div 10,x);
           x:=x*10+n mod 10
         end
end;
```

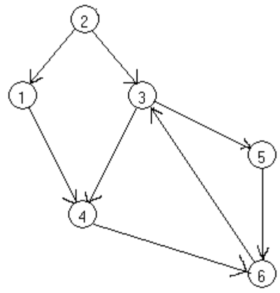
Limbajul C/C++

```
void f(int n, int &x)
{  if(n>0)
        if(n%2==1)
        {
            x=x*10+n%10;
            f(n/10,x);
        }
        else
        {
            f(n/10,x);
            x=x*10+n%10;
        }
}
```

a. 1078	b. 1087	c.7018	d. 7108

3. Utilizând metoda backtracking se generează toate combinările de câte 4 elemente ale mulţimii {1,2,3,4,5,6}. Dacă primele trei permutări generate sunt, în acestă ordine: 1234, 1235,1236 precizaţi care este ultima permutare și numărul de ordine al acesteia

a. 6543 și 15	b. 3456 și 15	c. 2345 și 720	d. 2346 și 30

4.	Fie graful orientat alăturat. Numărul de componente tare conexe de graf sunt: 			
	a.0	**b. 1**	**c. 2**	**d. 3**
5.	Fie G=(V,E) un graf neorientat astfel încât V=$A \cup B$, $A \cap B = \emptyset$, E=$\{(x, y) \mid x \in A, y \in B\}$. Stabiliți afirmația care este adevărată.			
	a. Graful este hamiltonian	**b.** Graful **NU** conține cicluri	**c.** Graful **G** este arbore	**d.** Graful G este complet

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte n,m (numere naturale)
┌cât timp n-m>1 execută
│ x←n-m
│ n←m
│ m←x
└■
┌Dacă m=3 și n=2 atunci
│     scrie 'Da'
│altfel
│     scrie 'Nu'
└■
```

a)	Scrieți ce va afișa algorimtul dacă se citesc valorile **89** și **55**?	**(6p.)**
b)	Scrieţi două seturi de numere naturale diferite, fiecare număr având **3** cifre, care pot fi citite pentru variabilele n și m astfel încât mesajul afișat să fie **DA**.	**(6p.)**
c)	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura cât timp...execută cu o structură repetitivă de alt tip.	**(6p.)**
d)	Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat.	**(10p.)**

2.	În declararea următoare, câmpurile **h**, **m** şi **s** ale înregistrării pot memora minutul şi secunda unui interval de timp. Scrieţi în limbajul **Pascal/C/C++** instrucțiunile necesare adunării a două intervale de timp memorate în variabilele **t1** și **t2** în variabila **t**.

Limbajul Pascal	Limbajul C/C++
`type timp=record` `h,m,s:integer` `end;` `var t1,t2,t:timp;`	`struct timp` `{` `int h,m,s;` `} t1,t2,t;`

Scrieţi în limbajul **Pascal /C/C++** instrucțiunile necesare adunării a două intervale de timp memorate în variabilele **t1** și **t2** în variabila **t.** **(6p.)**

3.	Variabila s poate memora un șir cu maximum **20** de caractere. Scrieți ce se afișează în urma executării secvenței de instrucțiuni de mai jos: **(6p.)**

Limbajul Pascal	Limbajul C/C++
`s:='3x-2=7';` `s:=copy(s,2,5);` `delete(s,2,2);` `s[2]:=chr(ord(s[2])+2);` `write(s);`	`char s[10]="3x-2=7";` `strcpy(s,s+1);` `strcpy(s+1,s+3);` `s[2]=s[2]+2;` `cout<<s; | printf("%s",s);`

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Scrieţi definiţia completă a unui subprogram **divizori**, cu trei parametri, care primeşte prin intermediul parametrilor:

- **a**, un număr natural cu cel mult 9 cifre;
- **n**, un număr natural nenul ($0 < n < 100$);
- **v**, un tablou unidimensional care memorează un şir de n numere naturale, fiecare având cel mult nouă cifre.

Subprogramul **divizori** determină memorarea în tabloul unidimensional v a tuturor divizorilor naturali ai parametrului a. Subprogramul va furniza prin intermediul parametrului n numărul de divizori naturali ai numărului a și prin parametrul v divizorii numărului a.

Exemplu. Pentru valorile **a=1000**, subprogramul va furniza tabloul unidimensional **v=(1 2 4 5 8 10 20 25 40 50 100 125 200 250 500 1000)** și **n=16**. (**10p**.)

2. Să se construiască o matrice pătratică cu n linii şi n coloane care conţine pe prima linie, de la stânga la dreapta toate numerele naturale de la **n** la **1**, în ordine descrescătoare, pe linia a doua, de la stânga la dreapta, toate numerele naturale de la **n-1** la **0**, în ordine descrescătoare, pe linia a treia, de la stânga la dreapta, toate numerele naturale de la **n-2** la **0**, în ordine descrescătoare, restul elementelor se vor completa cu **0**. Pe linia a **4**-a se vor completa de la stânga la dreapta toate numerele de la **n-3** la **0**, în ordine

	descrescătoare, restul elementelor se vor completa cu **0**, şi aşa mai departe până la linia n. Scrieţi programul **Pascal/C/C++** care citeşte de la tastatură valoarea naturală n (**2<n<25**), construieşte matricea conform cerinţei şi o afişează pe ecran, pe linii, cu spaţii între elementele de pe fiecare linie. De exemplu, dacă se citeşte de la tastatură **n=5**, se va construi şi se va afişa pe ecran matricea: **5 4 3 2 1** **4 3 2 1 0** **3 2 1 0 0** **2 1 0 0 0** **1 0 0 0 0** **(10p.)**
3.	Fişierul text **bac.in** conţine un şir **s** de cel mult un milion de cifre. a) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină și afișează pe ecran cifrele în ordinea descrescătoare a numărului de apariții urmate de numărul lor de apariții. **(8p.)** Exemplu: dacă fişierul **bac.in** are conţinutul: **2 3 4 5 5 3 2 2 2 4 5 1 1 1 3 4 5 2 3 4 5 5 5 3 4 3 3 3 3 2 2 3 2 3 2 3 2 3 4** atunci, pe ecran se va afişa **3 13 2 10 5 7 4 6 1 3** b) Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

Observaţii:

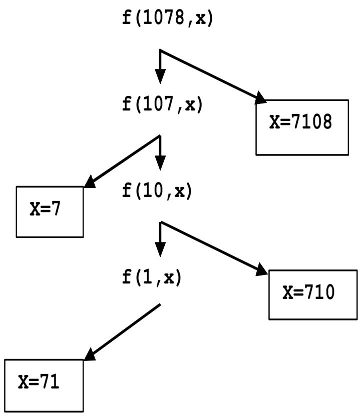
• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

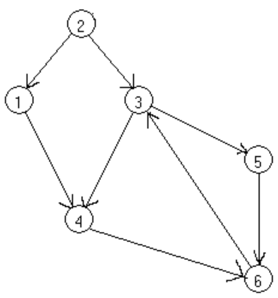
VARIANTA 17 - *Rezolvare*

Subiectul 1

<table>
<tr><td>1.</td><td colspan="2">Se observă că x mod 3 are valoarea 0,1 sau 2 în funcție de valoarea lui x. Dacă x este 0 sau este un multiplu a lui 3 atunci x mod 3 are valoare 0, iar (x+1) mod 3 are valoare 1 și (x+2) mod 3 are valoarea 2. Dacă x este de form 3k+1 atunci x mod 3 are valoare 1, (x+1) mod 3 are valoarea 2 și (x+2) mod 3 are valoarea 0. Dacă x este de form 3k+2 atunci x mod 3 are valoare 2, (x+1) mod 3 are valoarea 0 și (x+2) mod 3 are valoarea 1.

Pentru precizarea răspunsului b) se acordă 4p.</td></tr>
<tr><td>2.</td><td colspan="2">În desenul următor este prezentat mecanismul prin care se execută apelul f(1078, x) al subprogramului recursiv f.



Pentru răspunsul d) se acordă 4p.</td></tr>
<tr><td>3.</td><td colspan="2">Numărul cerut este: 3456 și are numărul de ordine 15.
Pentru răspunsul b) se acordă 4p</td></tr>
<tr><td>4.</td><td>Graful are o singură componentă tare conexă compusă din vârfurile: 3,4,5,6
Pentru răspunsul b) se acordă 4p.</td><td>
</td></tr>
<tr><td>5.</td><td colspan="2">Fie G=(V,E) un graf neorientat astfel încât V=$A \cup B$, $A \cap B = \emptyset$, E=$\{(x, y) | x \in A, y \in B\}$. Graful definit în enunț este graful este bipartit, în care fiecare muchie are un vârf într-o mulțime și un alt vârf într-o altă mulțime. Un graf bipartit nu are cicluri. Un graf bipartit poate să nu fie conex, nu poate fi graf hamiltonian, și nu este complet, deoarece nu există muchii care să fie incidente cu</td></tr>
</table>

	vârfuri care aparțin aceleaiași mulțimi. Pentru precizarea răspunsului **b)** se acordă **4p**.

Subiectul 2

<table>
<tr><td rowspan="4">1.</td><td>a)</td><td>Algoritmul verifică dacă cele două numere citite n și m sunt doi termeni consecutivi ai șirului lui Fibonacci cu n≥m.
Pentru precizarea mesajului DA se acordă 6p.</td></tr>
<tr><td>b)</td><td>Pentru seturile de valori n=233, m=144, respectiv n=610, m=377 se obține mesajul Da. Pentru identificarea unui set de valori se acordă 3p. În celelalte cazuri se acordă 0p</td></tr>
<tr><td>c)</td><td>Înlocuirea structurii <code>cât timp n>0 execută...</code> presupune utilizarea unei structuri repetitive repetă - cât timp sau repetă - până când:
<pre>
Citeste n,m
┌dacă n-m>1 atunci
│ ┌repetă
│ │ x←n-m
│ │ n←x
│ │ m←x
│ └cât timp n-m>1
└■
┌dacă n=3 și m=2 atunci
│ Scrie 'Da'
│ altfel
│ Scrie 'Nu'
└■
</pre>
<pre>
Citeste n,m
┌dacă n-m>1 atunci
│ ┌repetă
│ │ x←n-m
│ │ n←x
│ │ m←x
│ └până când n-m ≤ 1
└■
┌dacă n=3 și m=2 atunci
│ Scrie 'Da'
│ altfel
│ Scrie 'Nu'
└■
</pre>
Fără testarea inițială a valorii lui n-m (n-m>1), algoritmul obținut prin înlocuirea mecanică a structurii cât_timp cu structura repetă... va afișa pentru valorile inițiale n=3 și m=2 mesajul NU deși în algoritmul inițial se va afișa răspunsul Da. Pentru alegerea unei structuri repetitive se acordă 2p., pentru testarea inițială a valorii lui n se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 1p, iar pentru scrierea integrală a restului algoritmului se mai acordă 1p. În total 6p.</td></tr>
<tr><td>d)</td><td>Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc.

Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afișare a rezultatului și de decizie se acordă câte 1p., pentru instrucţiunea repetitivă se acordă 3p., iar pentru cele 6 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total 10p.</td></tr>
</table>

2.	**Un răspuns corect:** `Pascal: t.s:=(t1.s+t2.s) mod 60;` `         t.m:=t1.m+t2.m +(t1.s+t2.s) div 60;` `         t.h=t1.h+t2.h + (t1.m+t2.m) div 60;` `C/C++:   t.s=(t1.s+t2.s)%60;` `         t.m=t1.m+t2.m +(t1.s+t2.s)/60;` `         t.h=t1.h+t2.h + (t1.m+t1.m)/60;` Pentru orice răspuns corect se acordă **6p**.
3.	Răspunsul correct este **x=9**. Pentru răspuns corect **6p.**

Subiectul 3

1\. Prelucrarea oferită de subprogram presupune parcurgerea tuturor valorilor șirului de numere de la 1 la a/2 și adăugarea în tabloul v a valorilor care sunt divizori ai numărului a. La fiecare adăugare crește valoarea numărului n. La final se adaugă numărul a la tabloul unidimensional v.

Limbajul Pascal

```
Type vector=array[1..100]of longint;
procedure sub(a:longint; var v:vector; var n:byte);
var  i,j: longint;
begin
      i:=1;j:=0;
         for i:=1 to a div 2 do
            if a mod i=0 then
                 begin
                 j:=j+1;
                 v[j]:=i;
                end;
      j:=j+1;
      v[j]:=a;
n:=j;
end;
```

Limbajul C/C++

```
void divizori (int a, int v[100], int &n)
{int i=1, j=0;
for (i=1;i<=a/2;i++)
    if(a%i==0) v[++j]=i;
v[++j]=a;
n=j;
}
```

Pentru antet corect se acordă 3p. (Pentru respectarea structurii antetului (procedure/void) se acordă 1p., pentru declararea corectă a parametrilor de intare a și b încă 1p., pentru declararea corectă a parametrilor **n** și **v** încă 1p.).

Pentru declararea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p.

Pentru determinarea unui divizor al parametrului a 2p. Pentru determinarea numărului de divizori ai numărului a 2p. Pentru corectitudinea globală a programului 1p. În total **10p.**

2.	Se pot scrie mai mulți algoritmi pentru a obține a matricea cerută. Unul dintre algoritmi care completează coloanele matricei (numerotate de la **1** la **n**), respectiv liniile (numerotate de la **1** la **n**), în maniera descrisă, este: `┌pentru i←1,n execută` `│┌pentru j←1,n execută` `││ ┌dacă i+j ≤ n+1 atunci` `││ │  A[i,j]←n+2-i-j` `││ │altfel` `││ │  A[i,j]←0` `││ └■` `│└■` `└■` Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui n se mai acordă 1p., pentru parcurgerea unei coloane sau linii se acordă 1p., pentru completarea corectă a unei linii se acordă 2p., pentru completarea corectă a tuturor liniilor se acordă 2p., iar pentru afișarea matricei conform cerinței se acordă 2p. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă 1p. – în total **10p.**	
3.	**b)**	**Algoritmul de rezolvare propus** citește, unul câte unul, numerele din fișier, și le contorizează într-un vector de frecvență. Apoi ordonează descrescător vectorul după frecvența cifrelor. Algoritmul utilizează un alt vector pentru a reține cifrele în paralele cu frecvența lor. Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul. Deoarece numărul de valori din fișier (1000000) este dimensiunea care contează în problemă, spunem că am obținut un algoritm liniar. Deoarece se va ordona un vector cu 10 valori, numărul de iterații este de 90, se poate presupune că timpul de execuție este mai mic decât timpul necesar pentru citirea valorilor din fișier. Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. – în total **2p.**
	a)	**Limbajul Pascal**

```
var f:text;
    i,j,aux,x:integer;
    fr,v:array[0..9] of integer;
begin
  assign(f,'bac.in');reset(f);
for i:=0 to 9 do v[i]:=i;
while not eof(f) do begin
      read(f,x);
      fr[x]:=fr[x]+1;
      end;
for i:=0 to 9 do
  for j:=i+1 to 9 do
    if fr[i]<fr[j] then
      begin
         aux:= fr[i];
         fr[i]:=fr[j];
         fr[j]:=aux;
         aux:=v[i];
         v[i]:=v[j];
         v[j]:=aux;
      end;
```

```
for i:=0 to 9 do
  write(v[i],' ',fr[i],' ');
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int fr[10],v[10];

int main()
{
    int x,i,j,aux;
    ifstream f("bac.in");
    for(i=0;i<=9;i++)v[i]=i;
    while(f>>x)fr[x]++;
    for (i=0;i<9;i++)
        for(j=i+1;j<=9;j++)
    if(fr[i]<fr[j]){
        aux=fr[i];
        fr[i]=fr[j];
        fr[j]=aux;
        aux=v[i];
        v[i]=v[j];
        v[j]=aux;
    }

    for (i=0;i<=9;i++) cout<< v[i]<<' '<<fr[i]<<endl;

    return 0;
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea valorilor cerute, 1p. pentru afișarea ei, 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare (O(n)) și 1p. pentru utilizarea eficentă a memoriei. În total **8p.**

VARIANTA 18

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Variabilele **x, y** şi **z** sunt de tip întreg, **x=10**, **y=11, z=12**. Stabiliți rezultatul expresiei alăturate. **(4p.)**

Limbajul Pascal	Limbajul C/C++
`x div y+y div z+z div x`	`x/y+y/z+z/x`

a. 1	b. 1.3	c. 3	d. 33

2. Variabila **a** memorează elementele întregi ale unei matrice cu **10** linii şi **10** coloane numerotate de la **1** la **10**. Stabiliți ce afișează secvența de mai jos.(**4p**.)

Limbajul Pascal

```
for i:=1 to 10 do
  for j:=1 to 11-i do
     if a[i][j]>10 then
      Write(a[i][j],' ');
```

Limbajul C/C++

```
for(i=1;i<=10;i++)
   for(j=1;j<=11-i;j++)
     if(a[i][j]>10)
         cout<<a[i][j]<<" ";
         | printf("%d ", a[i][j]);
```

a. Elementele mai mari ca valoarea 10 aflate deasupra diagonalei principală	b. Elementele mai mari ca valoarea 10 aflate sub diagonala principală	c. Elementele mai mari ca valoarea 10 aflate deasupra diagonalei secundare	d. Elementele mai mari ca valoarea 10 aflate sub diagonala secundară

3. Utilizând metoda backtracking se generează toate modurile de a scrie numărul 9 ca sumă de numere naturale nenule distincte: **1+2+6**, **1+3+5**, **2+3+4**, **2+7**, **3+6**, **4+5**. Aplicând aceeași metodă pentru scrierea numărului 10 stabiliți numărul de soluții de forma 1+...(**4p**.)

a. 3	b. 4	c. 5	d. 6

4. Un graf neorientat cu **6** noduri, numerotate de la **1** la **6** are muchiile **[1,5]**, **[2,3]**, **[2,4]**, **[2,5]**, **[3,4]**, **[4,5]**, **[5,6]**. Stabiliți numărul de cicluri elementare distincte de lungime 3 existente în graf. Două cicluri sunt distincte dacă diferă prin cel puţin o muchie

a. 3	b. 4	c. 5	d. 6

5. Pentru reprezentarea unui arbore cu rădăcină cu **9** noduri, etichetate cu numere naturale de la **1** la **9**, se utilizează vectorul de "taţi": **T=(5,0,2,7,3,3,2,4,7)**. Lungimea maximă a unui lanţ elementar care leagă două noduri oarecare din acest arbore este:

a. 2	b. 4	c. 5	d. 6

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte a,b (numere naturale)
┌dacă a≥b atunci
│    c←a; a←b; b←c
└■
┌pentru x←a,b execută
│  c←x%10;ok←1; y←[x/10];
│  ┌cât timp y>0 execută
│  │ d←y%10
│  │ y←[y/10]
│  │ ┌dacă (c+d)%2=1 atunci
│  │ │    ok←0
│  │ └■
│  └■
│ ┌Dacă ok=1 atunci
│ │    scrie x
│ └■
└■
```

a) Ce valoare va fi afişată dacă se citesc valorile **311** şi **285**? **(6p.)**

b) Dacă pentru **a** se citește valoarea **20** scrieţi două valori care pot fi citite pentru **b** astfel încât să se afișeze **6** valori pentru fiecare dintre valorile citite. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **pentru...execută** cu o structură repetitivă de alt tip. **(6p.)**

d) Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. În declararea următoare, câmpurile **st** şi **dr** ale înregistrării memorează capătul din stânga respectiv capătul din dreapta ale unui interval închis.

Limbajul Pascal	Limbajul C/C++
`type interval=record` `st,dr:integer` `end;` `var A,B:interval;`	`struct interval` `{` `int st,dr;` `} A,B;`

Scrieţi în limbajul **Pascal/C/C++** instrucțiunile necesare afișării mesajului **se intersectează** dacă cele două intervale au valori comune sau mesajul **VID** dacă cele două intervale nu au valori commune. **(6p.)**

3.	Se consideră subprogramul **f** cu definiția următoare:	
	Limbajul Pascal	**Limbajul C/C++**

```
function
f(x,y:integer):integer;
begin
  if (y+1)*(y+1)<=x then
     begin
     f:=1+f(x,y+1)
     end
  else f:=1;
end;
```

```
 int f(int x, int y)
{if ((y+1)*(y+1)<=x)
      return 1+f(x,y+1);
  else return 1;
}
```

a) Scrieţi ce se va afişa în urma apelului **f(10,1).** **(3p.)**

b) Scrieţi trei valori pe care le poate avea variabilă x, astfel încât pentru apelul **f(x,1)** să se returneze **valoarea 8**. **(3p.)**

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Scrieţi definiţia completă a unui subprogram **suma**, cu doi parametri, care primeşte prin intermediul parametrilor:

– **n**, un număr natural nenul (**0 < n < 100**);
– **v**, un tablou unidimensional care memorează un şir de **n** numere naturale, fiecare având cel mult nouă cifre.

Subprogramul **suma**, afișează pe ecran, pe aceeași linie, separate prin câte un spațiu, acele numere din de tabloul **v** care au suma cifrelor aflate pe poziții pare egală cu suma cifrelor aflate pe poziții impare în număr. Într-un număr cifrele sunt numerotate de la dreapta la stânga astfel: cifra unităților se află pe poziția 0, cifra zecilor se află pe poziția 1, cifra sutelor pe poziția 2 etc.

Dacă tabloul nu conţine un astfel de număr, suprogramul va afișa mesajul **Nu există**.
Exemplu. Pentru **n=3** și **v=(1243, 110,1209, 5555)** subprogramul va afișa valorile:
1243 110 5555 **(10p.)**

2. Un subșir al unui șir de caractere este o secvență de caractere aflate pe poziții consecutive în acest șir. Se consideră un text cu cel mult 100 de caractere (doar litere mici ale alfabetului englez şi spaţii). Scrieţi un program **Pascal /C/C++** care citeşte de la tastatură un text de tipul menţionat mai sus şi care determină şi afişează pe ecran cea mai lungă secvență din șir care conține doar consoane. Dacă există două astfel de secvențe se va afișa prima dintre ele. **(10p.)**
Exemplu
Dacă se citește șirul ”**abcdbefaeoudafktrgsbdp**” se va afișa subșirul ”**fktrgsbdp**”

3. Fişierul text bac.in conţine un şir **s** de cel mult un milion de naturale nenule, formate fiecare din cel mult **9** cifre, separate prin câte un spaţiu.

a) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină și afișează pe ecran numărul de

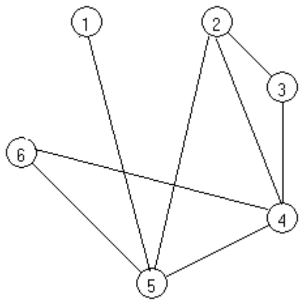
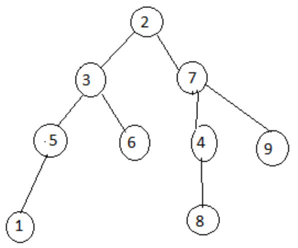
	secvențe strict crescătoare formate din câte **4** numere situate pe poziții consecutive în şir. O secvenţă a unui şir constă în elemente aflate pe poziţii consecutive în şirul considerat. **(8p.)** **Exemplu:** dacă fişierul bac.in are conţinutul: **10 11 12 13 17 20 19 24 25 26 27 5 6 4 3 11 11 12 13 14 15 16 34** atunci, pe ecran se va afişa 5 **b)** Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

⚠ **Observaţii:**

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 18 - *Rezolvare*

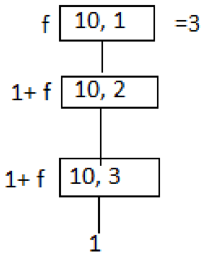
Subiectul 1

<table>
<tr><td>1.</td><td colspan="2">Pentru precizarea răspunsului c) se acordă 4p.</td></tr>
<tr><td>2.</td><td colspan="2">Pentru răspunsul c) se acordă 4p.</td></tr>
<tr><td>3.</td><td colspan="2">Soluțiile cerute sunt: 1+2+3+4, 1+2+7, 1+3+6, 1+4+5, 1+9. Sunt generate 5 numere cu proprietatea din enunţ..
Pentru răspunsul c) se acordă 4p.</td></tr>
<tr><td rowspan="2">4.</td><td colspan="2">Pentru precizarea răspunsului a) se acordă 4p.</td></tr>
<tr><td></td><td>Ciclurile elementare distincte de lungime 3 din graf sunt:
2, 3, 4, 2
2, 4, 5, 2
5, 6, 4, 5</td></tr>
<tr><td rowspan="2">5.</td><td colspan="2">Pentru precizarea răspunsului d) se acordă 4p.</td></tr>
<tr><td>Cel mai lung lanț elementar are lungimea: 6 și este compus din nodurile: 1, 5, 3, 2, 7, 4, 8</td><td></td></tr>
</table>

Subiectul 2

<table>
<tr><td rowspan="3">1.</td><td>a)</td><td>Algoritmul afișează numerele cuprinse în intervalul [a,b] în care toate cifrele au aceeași paritate. Se vor afișa valorile: 286, 288, 311
Pentru precizarea valorilor corecte se acordă 6p.</td></tr>
<tr><td>b)</td><td>Răspuns corect 6p. Un răspuns corect poate fi compus din două dintre numerele 10, 11, 31, 32</td></tr>
<tr><td>c)</td><td>Înlocuirea structurii <code>pentru... execută</code> presupune utilizarea unei structuri repetitive cât timp… execută sau repetă - până când:</td></tr>
</table>

		<pre>citeşte a,b (numere naturale) ┌dacă a≥b atunci │ c←a; a←b; b←c └■ x←a ┌cât timp x≤b execută │ c←x%10;ok←1; y←[x/10]; │ ┌cât timp y>0 execută │ │ d←y%10 │ │ y←[y/10] │ │ ┌dacă (c+d)%2=1 atunci │ │ │ ok←0 │ │ └■ │ └■ │ ┌Dacă ok=1 atunci │ │ scrie x │ └■ │ x←x+1 └■</pre> Pentru alegerea unei structuri repetitive se acordă 2p., pentru valoarea inițială a valorii lui **x** se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 1p, iar pentru scrierea integrală a restului algoritmului se mai acordă 1p. - în total **6p**.
	d)	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc. Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afișare a rezultatului și de decizie se acordă câte 1p., pentru instrucţiunea repetitivă se acordă 3p., iar pentru cele 6 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.
2.	**Un răspuns corect:** <pre>Pascal: if A.dr < B.st or B.dr < A.st then Write (”VID”); Else write (”se intersecteaza”); C/C++: if (A.dr < B.st || B.dr < A.st) cout<<”VID”; else cout<<”se intersecteaza”;</pre> Pentru orice răspuns corect se acordă **6p**.	
3.	În desenul următor este prezentat mecanismul prin care se execută apelul **f(10, 1)** al subprogramului recursiv **f**. Se va afişa, valoarea 3**,** răspuns pentru care se acordă **3p**.	



Funcția calculează partea întreagă a rădăcinii pătrate a numărului x. Pentru valoarea 8 se poate lua orice valoare din intervalul [**64, 80**] **3p.**

Subiectul 3

3. Prelucrarea oferită de subprogram presupune parcurgerea tuturor valorilor din tablou și verificarea proprietății cerute.

Limbajul Pascal

```
Type vector=array[1..100]of longint;
procedure cifre(n:integer; v:vector);
var  i,j,k,sp,si, gasit: longint;
begin
      gasit:=0;
         for i:=1 to n do
           begin
            x:=v[i];k:=0;
            while x>0 do begin
                          x:=x div 10; k:=k+1;
                         end;
             x:=v[i];
             sp:=0;si:=0;
             while x>0 do
                      begin
                        if k%2<> 0 then
                          si:=si+x mod 10
                        else sp:=sp+x mod 10;
                        x: =x div 10;
                        k:=k-1;
                      end;
        if sp=si then
           begin
            write(v[i],” “); gasit:=1;
           end;
       end;
     if gasit=0 then write(”Nu exista”);
end;
```

Limbajul C/C++

```
void cifre (int n, int v[100])
{ int i=1, j=0,k, sp,si, gasit=0;
  for (i=1;i<=n;i++)
  { x=v[i];k=0;
    while(x>0) {x=x/10; k++;}
    x=v[i];
    sp=0;
    si=0;
```

```
    while(x>0)
    { if (k%2!=0)
            si=si+x%10;
      else
            sp=sp+x%10;
      x=x/10;
      k--;
    }
    if (sp==si ){ cout<<v[i]<<" "; gasit=1;}
  }
  if (gasit==0) cout<<"Nu exista";
}
```

Pentru antet corect se acordă **3p**. (Pentru respectarea structurii antetului (**procedure/void**) se acordă **1p**., pentru declararea corectă a parametrilor de intare a și b încă 1p., pentru declararea corectă a parametrilor **n** și **v** încă 1p.).

Pentru declararea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p.

Pentru determinarea unui număr 3p. Pentru afișarea mesajului Nu există 1p.

Pentru corectitudinea globală a programului 1p.

În total **10p.**

Limbajul Pascal

```
Var s:string;   i,n,j,nr,lmax,pozj:integer;
begin
 read(s);
 lmax:=0;
 i:=1;
 n:=length(s);
 while s[i] in ['a','e','i','o','u'] do  i:=i+1;
 nr:=1; j:=i+1;
 while j<=n do
  if not(s[j] in ['a','e','i','o','u']) then
    begin
     nr:=nr+1; j:=j+1;
    end
  else
    begin
     if lmax<nr then
      begin
       lmax:=nr; pozj:=j-1;
      end;
     nr:=0; j:=j+1;
    end;
 if lmax<nr then
   begin
    lmax:=nr; pozj:=j-1;
   end;
 for i:=pozj-lmax+1 to pozj do write(s[i]);
end.
```

	Limbajul C++ `#include <iostream>` `#include<cstring>` `using namespace std;` `int main()` `{   char s[100],voc[6]="aeiou";` `    int i,j,pozi,pozj,lmax=0,nr;` `    cin.getline(s,100);` `    i=0;` `while(strchr(voc,s[i])!=NULL)i++;` `    nr=1;j=i+1;` `    while (j<strlen(s))` `    if (strchr(voc,s[j])==NULL) {nr++;j++;}` `    else` `    { if(lmax<nr)` `     {lmax=nr; pozj=j-1;}` `            nr=0; j++;}` `    if(lmax<nr){lmax=nr; pozj=j-1;}` `    for (i=pozj-lmax+1;i<=pozj;i++)cout<<s[i];` `    return 0;` `}` Pentru declararea corectă a sirului de caractere se acordă 1p., pentru citirea sirului se mai acordă 1p., pentru identificarea consoanelor se acordă un 1p., pentru numărarea consoanelor dintr-un şir se acordă 2p., pentru identificarea celei mai lungi secvente de consoane 2p., iar pentru afişarea secvenţei conform cerinţei se acordă 2p. În plus, pentru declararea variabilelor simple, structura şi corectitudinea sintactică a programului se mai acordă 1p. – în total **10p**.	
4.	**b)**	**Algoritmul de rezolvare propus** citeşte, primele patru numere şi verifică proprietatea cerute. Apoi citeşte număr cu număr, analizând pentru noile numere citite proprietatea cerută. Eficienţa algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată şirul. Deoarece numărul de valori din fişier (1000000) este dimensiunea care contează în problemă, spunem că am obţinut un algoritm liniar. Ca spaţiu de memorie, soluţia propusă este eficientă, deoarece utilizează doar variabile simple. Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienţei, încă 1p. – în total **2p.**
	a)	**Limbajul Pascal** `var f:text;` `    x,y,z,t:longint;` `begin` `  assign(f,'bac.in');reset(f);` `   read(f,x); read(f,y); read(f,z); read(f,t);` `   nr:=0;` `  if x<y and y<z and z<t then  nr:=nr+1;` `  while not eof(f) do begin` `      x:=y;` `      y:=z;` `      z:=t;` `      read(f,t);` `      if x<y and y<z and z<t then  nr:=nr+1;` `      end;` `    close(f);` `  write(nr);` `end.`

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{ int x,y,z,t,nr=0;
  ifstream f("aparitii.in);
  f>>x>>y>>z>>t;
  if(x<y && y<z && t<z) nr++;
  while(!f.eof())
  { x=y;
    y=z;
    z=t
    f>>t;
    if(x<y && y<z t<z) nr++;
  }
  cout<<nr;
  return 0;
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea valorii cerute, 1p. pentru afișarea ei, 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare (O(n)) și 1p. pentru utilizarea eficentă a memoriei. În total **8p.**

VARIANTA 19

SUBIECTUL I **(20 de puncte)**

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Indicaţi expresia care are valoarea **true/1** dacă şi numai dacă valoarea variabilei întregi **a** este negativă şi impară.

Limbajul Pascal	Limbajul C/C++
a. `(a<0) or (a mod 2 = 1)`	**a.** `(a<0) || ( a % 2 == 1)`
b. `(a>0) and (a mod 2 = 1)`	**b.** `(a>0) && (a % 2 == 1)`
c. `a mod 2 = -1`	**c.** `a % 2 == -1`
d. `(a<0) and (a mod 2 = 1)`	**d.** `(a<0) && (a % 2 == 1)`

2. O fracţie echiunitară este o fracţie care are numărătorul egal cu numitorul.
Se consideră tipul de date **fractie**, capabil să memoreze numărătorul şi njumitorul unei fracţii. Indicaţi expresia care are valoarea **true/1** dacă şi numai dacă variabila **f** de tipul **fractie** este echiunitară.

Limbajul Pascal

```
type  fractie=record
          x,y:integer
      end;
var f:fractie;
```

a. `x.f=y.f` **b.** `y.x=x.y`
c. `f.x=f.y` **d.** `x.x=y.y`

Limbajul C/C++

```
struct fractie {
      int x,y;
};
fractie f;
```

a. `x.f==y.f` **b.** `y.x==x.y`
c. `f.x==f.y` **d.** `x.x==y.y`

3. Utilizând metoda backtracking, se generează toate posibilităţile de a forma succesiuni de câte **5** rechizite distincte din mulţimea {**creion**, **pix**, **stilou**, **gumă**, **echer**}, astfel încât în fiecare succesiune stiloul se aşază înaintea gumei. Două succesiuni sunt distincte dacă rechizitele sunt în altă ordine.
Primele cinci soluţii generate sunt, în această ordine:
{**creion**, **pix**, **stilou**, **gumă**, **echer**}, {**creion**, **pix**, **stilou**, **echer**, **gumă**}, {**creion**, **pix**, **echer**, **stilou**, **gumă**}, {**creion**, **stilou**, **pix**, **gumă**, **echer**}, {**creion**, **stilou**, **pix**, **echer**, **gumă**}
Indicaţi soluţia generată imediat înainte de {**echer**, **stilou**, **gumă**, **creion**, **pix**}

a. {**pix**, **creion**, **gumă**, **stilou**, **echer**} **b.** {**pix**, **creion**, **stilou**, **gumă**, **echer**}
c. {**echer**, **stilou**, **pix**, **gumă**, **creion**} **d.** {**echer**, **pix**, **stilou**, **gumă**, **creion**}

4. Într-un arbore cu rădăcină se consideră că un nod se află pe nivelul **x** dacă lanţul elementar care are o extremitate în nodul respectiv şi cealaltă extremitate în rădăcina arborelui are lungimea **x**. Pe nivelul **0** se află un singur nod (rădăcina).
Un arbore cu rădăcină are **8** noduri şi **4** niveluri. Toate nodurile de pe acelaşi nivel (cu excepţia ultimului nivel) au un număr egal (nenul) de descendenţi direcţi (fii) şi există doar două niveluri cu acelaşi număr de noduri, celelalte niveluri au număr diferit de noduri. Indicaţi numărul minim de noduri de pe nivelul **1**.

a. 0 **b.** 1 **c.** 2 **d.** 4

5. Un graf neorientat are **6** noduri, numerotate de la **1** la **6** şi muchiile: [**1**,**5**], [**2**,**3**], [**2**,**5**], [**2**,**6**], [**3**,**4**], [**3**,**5**], [**4**,**5**]. Indicaţi numărul ciclurilor distincte ale acestui graf. Două cicluri sunt distincte dacă ele diferă prin cel puţin o muchie.

a. 1 **b.** 2 **c.** 3 **d.** 4

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Algoritmul alăturat este reprezentat în pseudocod.**
S-a notat cu **a%b** restul împărţirii numărului natural **a** la numărul natural nenul **b** şi cu **[c]** partea întreagă a numărului real **c**.

```
citeşte x,y  (numere naturale nenule, x<y)
a←x+y
i←2
┌repetă
│ j=0;
│┌cât timp a%i=0 execută
││  j←j+1
││ a←[a/i]
│└■
│┌dacă j%2≠0 atunci
││ scrie 'nu'
││ a←0
││altfel
││ i←i+1
│└■
└până când a≤1
┌dacă a=1 atunci
│ scrie 'da'
└■
```

a. Scrieţi mesajul care se afișează dacă se citesc, în această ordine, valorile **260** şi **1000**. **(6p.)**

b. Scrieţi patru perechi de numere consecutive de cel mult două cifre fiecare ce pot fi citite pentru **x** şi **y** (**x<y**) astfel încât, în urma executării algoritmului, să se afişeze **da**. **(6p.)**

c. Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

d. Scrieţi în pseudocod un algoritm, echivalent cu cel dat, care să nu conţină nicio structură repetitivă. **(6p.)**

2. În secvenţa de mai jos variabilele **i** şi **j** sunt de tip întreg şi variabila **A** este de tip tablou bidimensional cu **5** linii şi **5** coloane, numerotate de la **1** la **5**, cu elemente numere întregi. Fără a utiliza alte variabile, scrieţi o instrucţiune care poate înlocui punctele de suspensie astfel încât, în urma executării secvenţei obţinute, tabloul memorat în variabila **A** să aibă elementele din figura alăturată. **(6p.)**

```
1 0 0 0 0
1 2 1 0 0
0 2 3 2 0
0 0 3 4 3
0 0 0 4 5
```

Limbajul Pascal	Limbajul C/C++
`for i:=1 to 5 do`	`for(i=1;i<=5;i++)`
`  for j:=1 to 5 do`	`  for(j=1;j<=5;j++)`
`.............`	`.............`

3. Subprogramul **f** este definit mai jos.
a. Scrieți valoarea obţinută în urma apelului **f(5)**.
b. Scrieţi cea mai mică valoare de **4** cifre pe care o poate avea o variabilă **n**, astfel încât, pentru apelul **f(n)** să se obţină valoarea **6**. **(6p.)**

Limbajul Pascal

```
function f(x:integer):integer;
begin
     if x=0 then f:=0
     else
        if x mod 10 = 5 then
        f:=1+f(x div 10)
        else
           f:=2+f(x div 10)
end;
```

Limbajul C/C++

```
int f(int x)
{
    if(x==0) return 0;
    if(x%10==5)
       return 1+f(x/10);
    else
       return 2+f(x/10);
}
```

SUBIECTUL al III-lea **(30 de puncte)**

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Subprogramul **eliminare** are doi parametri**:**
- **n,** prin care primeşte câte un număr natural cu cifre nenule ($10^2 \leq n < 10^9$);
- **x**, prin care furnizează numărul natural obţinut din **n** prin eliminarea primei şi ultimei cifre.

Scrieți definiția completă a subprogramului.
Exemplu: pentru **n=123456**, după apel **x** va fi **2345**. **(10p.)**

2. Se consideră un text cu cel mult **250** de caractere (litere mici ale alfabetului englez şi spaţii), în care cuvintele sunt separate printr-un spaţiu. Scrieţi un program Pascal/C/C++ care citeşte de la tastatură un text de tipul menţionat mai sus şi determină transformarea acestuia în memorie prin modificarea tuturor cuvintelor de cel puţin trei litere astfel încât primul caracter să ocupe poziţia ultimului caracter din cuvânt, iar ultimul carcater să ocupe poziţia primului caracter din cuvânt. Programul afişează pe ecran textul modificat.
Exemplu: pentru textul **o proba si doua teste**
se obţine textul **o arobp si aoud eestt** **(10p.)**

3. Fişierul **bac.in** conţine pe prima linie cel mult 10^6 numere naturale nenule mai mici decât **100**, separate prin câte un spaţiu.
Se cere să se afişeze pe ecran **cel mai mic** număr **x** care verifică fiecare dintre următoarele condiţii:
- **x** nu apare în fişier;
- există un număr **y** în fişier astfel încât **x+y=100**.

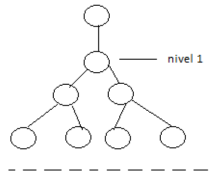
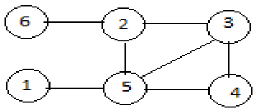
Dacă nu există niciun asemenea număr, se afişează mesajul **nu există**.
Proiectați un algoritm eficient din punctul de vedere al timpului de executare.
Exemplu: dacă fişierul conţine numerele
56 12 1 97 2 98 9 91 3 43 97 89 6
pe ecran se afişează **11**
a. Scrieți programul Pascal/C/C++ corespunzător algoritmului proiectat. **(8p.)**
b. Descrieți în limbaj natural algoritmul proiectat, justificând eficiența acestuia. **(2p.)**

Observaţii:

- Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
- Timpul de lucru efectiv este de 3 ore.
- Identificatorii utilizaţi în rezolvări trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată). Datele de intrare se consideră corecte, validarea lor nefiind necesară.
- În grafurile din cerințe oricare arc/muchie are extremități distincte și oricare două arce/muchii diferă prin cel puțin una dintre extremități.

VARIANTA 19 - *Rezolvare*

SUBIECTUL I

1.	Rezultatul expresiei **`x mod 5`** (Pascal) sau **`x%5`** (C/C++) este negativ în cazul unui operand **`x`** întreg negativ. Pentru precizarea răspunsului **a)** se acordă **4p**.
2.	Pentru răspunsul **c)** se acordă **4p.**
3.	Algoritmul de generare este cel al permutărilor, suplimentar verificându-se condiţia impusă. Pentru răspunsul **c)** se acordă **4p**.
4.	Cele două niveluri care au același număr de noduri sunt 0 și 1. Prin reprezentarea grafică a arborelui se observă că pe nivelul 1 avem un singur nod. Pentru precizarea răspunsului **b)** se acordă **4p**. 
5.	Graful dat se poate reprezenta grafic ca în figură.  Ciclurile elementare distincte sunt: **2 3 5 2** **3 4 5 3** **2 3 4 5 2** Pentru precizarea răspunsului **c)** se acordă **4p**.

SUBIECTUL al II-lea

1.	**a.**	Se determină exponenții fiecărui factor prim din descompunerea lui **a**. Se afișează mesajul **nu** dacă există cel puțin un exponent impar, alfel se afișează mesajul **da**. Altfel spus, se afișează mesajul **da** dacă numărul a este pătrat perfect, alfel se afișează mesajul **nu**. Pentru precizarea mesajului **nu** se acordă **6p**.	
	b.	Suma a două numere consecutive este un număr impar. Se determină numerele impare care au suma un pătrat perfect. Pentru aceasta vom considera toate pătratele numerelor impare: 3^2=9=4+5, 5^2=25=12+13, 7^2=49=24+25, 3^4=81=40+41, 11^2=121=60+61, 13^2=169=84+85. Pentru o pereche corectă se acordă 2p, pentru 2-3 perechi corecte se acordă 4p, iar pentru oricare patru perechi dintre (4,5), (12,13), (24,25), (40,41), (60,61), (84,85) și nicio pereche în plus, se acordă **6p**.	
	c.	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, experiențe care au cimentat cunoștințele privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor etc. Pentru instrucțiunile corecte de citire a datelor, de afişare a rezultatului se acordă câte 1p., pentru cele două instrucțiuni de decizie se acordă 2p., pentru cele două instrucţiuni repetitive se acordă 4p., iar pentru cele 7 atribuiri se acordă 1p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.	
	d.	Se verifică dacă numărul **a** este pătrat perfect. Pentru orice rezolvare corectă fără structuri repetitive sau recursive se acordă **6p**.	**citeşte x,y** (numere naturale nenule, **x<y**) **a←x+y** ┌**dacă** $\sqrt{a} \neq [\sqrt{a}]$ **atunci** │ **scrie 'nu'** │**altfel** │ **scrie 'da'** └■

2.

Descriem două posibile variante de rezolvare. Pe diagonala principală a tabloului A[i][i] se memorează i, iar pe „pseudodiagonalelele” din vecinătatea diagonalei principale (A[i][i-1] și A[i][i+1]) se memorează i-1 (i-|(i-j)|).

```
┌dacă |(i-j)|≤1  atunci
│ a[i,j]←i-|(i-j)|
│altfel
│ a[i,j]←0
└■
```

```
┌dacă i=j  atunci
│ a[i,j]←i
│altfel
│┌dacă |(i-j)|=1  atunci
││ a[i,j]←i-1
││altfel
││ a[i,j]←0
│└■
└■
```

Pentru adresarea corectă a unui element al tabloului se acordă 1p., pentru completarea corectă a diagonalei principale se acordă 1p., pentru completarea corectă a elementelor egale cu 0 se acordă 1p., pentru completarea corectă a celorlalte elemente se acordă 2p.,iar pentru corectitudinea sintactică a secvenței se mai acordă 1p. – în total **6p**.

3.

a. Se urmăresc apelurile recursive și se fac calculele:

```
f(5)= f(0) + 1
        └─►=0
```

f(5)=1
Pentru răspunsul 1 se acordă **3p**.

b. Numărul trebuie să fie format din două cifre 5 și două cifre diferite de 5. Cel mai mic astfel de număr cu patru cifre este 1055.
Pentru răspunsul 1055 se acordă **3p**.

SUBIECTUL al III-lea

1.

Prelucrarea oferită de subprogram construiește valoarea cerută prin “alipirea” la numărul x (inițial egal cu 0) a cifrelor lui n. Se îndepărtează ultima cifră. Deoarece cifrele lui n sunt accesibile de la ultima cifră către prima, “alipirea” se va realiza corect dacă se utilizează înmulțiri cu puteri consecutive ale lui 10, cât timp n are cel puțin două cifre.

Limbajul Pascal

```
procedure eliminare(n:longint; var x:longint);
var p:longint;
begin
        n:=n div 10;
        p:=1;
        while n>9 do
        begin
                x:=x + n mod 10 * p;
                p:=p * 10;
                n:= n div 10
        end
end;
```

Limbajul C/C++

```
void eliminare(int n, int &x)
{
    n=n/10;
    int p=1;
    while(n>9)
    {
        x=x+n%10*p;
        p*=10;
        n/=10;
    }
}
```

	Pentru respectarea structurii antetului (procedure/void) se acordă 1p., pentru declararea corectă a parametrului de intare n încă 1p., pentru declararea corectă a parametrului de ieșire x încă 1p., pentru declararea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p. Corectitudinea algoritmică a prelucrării în vederea obținerii valorii cerute este notată cu 4p. Pentru transmiterea rezultatului prin parametrul x se acordă 1p., aceasta însemnând că valoarea lui x nu se afișează și nu se returnează explicit. În total **10p.** Facem o primă observație: subprogramul **trebuie** să se numească eliminare. O altă observație se referă la faptul că, în acest caz, în C/C++, definirea subprogramuluii ca funcție cu rezultatul int (sau un alt tip în afară de void), deși posibilă, nu este logică și deci nu o recomandăm.
2.	Problema poate fi rezolvată în mai multe moduri.

Descriem acum o posibilă variantă de rezolvare. Se parcurge şirul şi se reţin în aux1 și aux2 pozițiile capetelor fiecărui cuvânt, poziții identificate prin căutarea spațiilor succesive din text. Pentru a separa mai uşor cuvintele, se adaugă la sfârşitul şirului un caracter spaţiu.

Limbajul Pascal

```
var s:string;
auxc:char;
i,j,aux1,aux2:integer;
begin
read(s);
s:=' '+s+' ';
i:=1;
while i<>length(s) do
begin
 if s[i]=' ' then
 begin
 aux1:=i+1; j:=i+1;
 while s[j]<>' ' do
 j:=j+1;
 aux2:=j-1;
 if (aux2-aux1>=2) then
 begin
  auxc:=s[aux1];
  s[aux1]:=s[aux2];
  s[aux2]:=auxc;
 end;
 i:=j;
 end;
 end;
delete(s,1,1);
delete(s,length(s),1);
writeln(s);
end.
```

Limbajul C/C++

```
#include <iostream>
#include <string.h>
using namespace std;
char s[251],sp[]=" ";
int main() {
char auxc;
int i,j,aux1,aux2;
cin.get(s,251);
strcpy(s,strcat(sp,s));
strcat(s," ");
i=0;
while(i<strlen(s)) {
  if(s[i]==' ') {
    aux1=i+1; j=i+1;
    while(s[j]!=' ')
       j++;
    aux2=j-1;
    if(aux2-aux1>=2) {
      auxc=s[aux1];
      s[aux1]=s[aux2];
      s[aux2]=auxc; }
     i=j;
   }
}
   strcpy(s,s+1);
   s[strlen(s)-1]='\0';
   cout<<s;
   return 0;
}
```

Sau, folosind funcții predefinite în limbaj:

Limbajul Pascal

```
var s:string;
    auxc:char;aux1,aux2:integer;
begin
  readln(s);
  s:=s+' ';
  aux1:=1;
  while aux1<=length(s) do
  begin
      aux2:=pos(' ',copy(s,aux1,255))-1;
```

```
      if (aux2>2) then
      begin
        auxc:=s[aux1];
        s[aux1]:=s[aux1+aux2-1];
        s[aux1+aux2-1]:=auxc;
      end;
      aux1:=aux1+aux2+1
  end;
  delete(s,length(s),1);
  writeln(s);
end.
```

Limbajul C/C++

```
#include <iostream>
#include <cstring>
using namespace std;
char s[251];
int main() {
  char auxc;
  char *aux1,*aux2;
  cin.get(s,251);
  strcat(s," ");
  aux1=s;
  while(strlen(aux1)>0) {
    aux2=strchr(aux1,' ')-1;
    if(aux2-aux1>=2) {
      auxc=*aux1;
      *aux1=*aux2;
      *aux2=auxc;
    }
    aux1=aux2+2;
  }
  s[strlen(s)-1]='\0';
  cout<<s;
  return 0;
}
```

Pentru declarare şir se acordă **1p.**, citire şir **1p.**, determinare cuvinte **2p.**, lungime cuvânt **1p.**, interschimbare caractere **2p.**, afișarea șirului de caractere cerut **2p.**, iar pentru corectitudinea sintactică **1p.** – în total **10p.**

3. **a.**

Limbajul Pascal

```
var f:text;
    x,sem:byte;
    apar:array[1..99] of byte;

begin
        assign(f,'date.in');reset(f);
        while not eof(f) do
        begin
                read(f,x); apar[x]:=1
        end;
        x:=1; sem:=0;
        while (sem=0) and (x<=99) do
```

```
         begin
            if (apar[100-x]=1) and (apar[x]=0) then
              begin
                 write(x);
                 sem:=1
              end;
                 inc(x)
         end;
close(f)
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int apar[100],x,gasit;
int main()
{
    while(f>>x) apar[x]=1;
    gasit=0;
    x=1;
    while(!gasit && x<=99)    {
        if(!apar[x] && apar[100-x])
        {
            cout<<x;
            gasit=1;
        }
        x++;
    }
    if(!gasit) cout<<"nu exista";
    f.close();
    return 0;
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 2p., pentru determinarea valorii cerute, 1p., pentru afișarea ei și 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor etc.). Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare (O(n)). În total **8p.**

Menționăm că prin algoritm principial corect se înțelege un algoritm care, în intenție, urmărește să rezolve corect problema, deși poate să prezinte scăpări logice sau confuzii.

b. O soluție posibilă:

Vom utiliza un vector cu ajutorul căruia marcăm aparițiile numerelor din fișier, apar, cu indicii între 1 și 99, vector care are inițial toate elementele egale cu 0.
Vom parcurge fișierul și, pentru fiecare valoare x citită, vom marca în vector apar[x]=1.
Căutăm primul element care nu a fost marcat (valoarea x pentru care apar[x]=0) și care are corespondent (valoarea 100-x) marcat (apar[100-x]=1).

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul cu un număr impresionant de numere. Deoarece numărul de valori din fișier este dimensiunea care contează în problemă, spunem că am obținut un algoritm liniar.
Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. – în total **2p.**

VARIANTA 20

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Știind că variabilele a și b memorează numere naturale cu a≤b.
Stabiliți care dintre expresiile **Pascal/C/C++** următoare indică numărul de numere pare din intervalul [a,b]:

	Limbajul Pascal	Limbajul C/C++
a.	`(b-a)div 2`	`(b-a)/2`
b.	`(b-a)div 2+1`	`(b-a)/2+1`
c.	`(b-a)div 2-(a mod 2)*(b mod 2)`	`(b-a)/2-(a%2)*(b%2)`
d.	`(b-a)div 2+1-(a mpd 2)*(b mod 2)`	`(b-a)/2+1-(a%2)*(b%2)`

2. Pentru subprogramul `f` cu definiția următoare, indicați valoarea care se va afișa în urma apelului `f(12345)`.

Limbajul Pascal

```
procedure f( x:integer);
begin
if x<>0 then
  if x mod 2=0 then begin
      write(x mod 10);
      f(x div 10);
      write(x mod 1   end
   else begin f(x div 10);
              write(x mod 10) end;
end;
```

Limbajul C/C++

```
void f(int x)
{
  if(x!=0)
    if(x%2==0) {
          cout<<x%10;
          f(x/10);
          cout<<x%10; }
     else
     { f(x/10); cout<<x%10;}
}
```

a. 54423231	b. 45432321	c. 4212345	d. 5432124

3. Utilizând metoda backtracking se generează toate submulțimile nevide ale mulțimii `{1,2,3,4}`, Primele cinci submulțimi generate sunt, în această ordine: `{1}`, `{1,2}`, `{1,2,3}`, `{1,2,3,4}`, `{1,2,4}`. Indicați care este cea de a **8**-a submulțime generată:

a. {1,3}	b. {1,3,4}	c. {1,4}	d. {2,3,4}

4. Se consideră arborele cu 10 noduri, numerotate de la `1` la `10` dat prin următorul vector "de taţi": `(0,1,2,3,1,5,5,7,5,9)`. Stabiliți numărul nodurilor care au exact `3` ascendenti.

a.0	b. 2	c. 3	d. 4

5. Care este numărul subgrafurilor complete ale grafului neorientat cu **5** noduri numerotate de la `1` la **5**, dat prin listele de adiacență alăturate?

```
1:2;
2:1,3,4;
3:2,4,5;
4:2,3,5;
5:3,4;
```

a. 0	b. 2	c. 8	d. 13

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul alăturat reprezentat în pseudocod:**
S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citește a,b (numere naturale)
p←1
┌cât timp a*b>0 și a%10=b%10 execută
| a←[a/10]
| b←[b/10]
| p←p*10
└■
a←a*p
scrie a
```

a) Ce valoare va fi afişată dacă se citesc valorile **12345** și **145**? **(6p.)**

b) Știind că pentru variabila **b** s-a citit valoarea **89**, scrieţi care este numărul numerelor cu patru cifre distincte, ce pot fi citite pentru variabila **a** astfel încât, pentru fiecare astfel de valoare, să se afişeze valoarea **50**. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **cât timp...execută** cu o structură repetitivă de alt tip. **(6p.)**

d) Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. Se consideră declarările următoare în care câmpurile **x** şi **y** ale înregistrării pot memora numere naturale ce reprezintă, în ordine, extremitatea inițială, respectiv extremitatea finală, a unui arc dintr-un graf orientat. Scrieţi o expresie în limbajul **Pascal|C/C++** care să aibă valoarea **true/1** dacă şi numai dacă arcele distincte **a1** şi **a2** sunt incidente. **(6p.)**

Limbajul Pascal

```
type arc=record
   x,y:integer
end;
var a1,a2: arc;
```

Limbajul C/C++

```
struct arc {
        int x,y;
};
arc a1, a2;
```

3. În secvenţa următoare, variabilele **i** şi **j** sunt de tip integer **int**, iar variabila **M** memorează o matrice cu **5** linii şi **5** coloane (numerotate de la **1** la **5**) cu elemente de tip **integer/int**. Scrieți care este cea mai mare valoare nemorată în matrice, la finalul executării secvenței, și care sunt coordonatele elementului din matrice (linia și coloana) ce memorează această valoare **(6p)**

Limbajul Pascal

```
x:=0;
for i:= 1 to 5 do
 for j:= 1 to 5 do begin
   if i mod 2 <> j mod 2
        then x:=x+2
        else dec(x);
   M[i,j]:=x
end
```

Limbajul C/C++

```
x=0;
for(i=1;i<=5;i++)
  for(j=1;j<=5;j++) {
     if ( i%2 != j%2 )
           x=x+2;
     else x--;
  M[i][j]=x;
}
```

Subiectul 3 (30p)

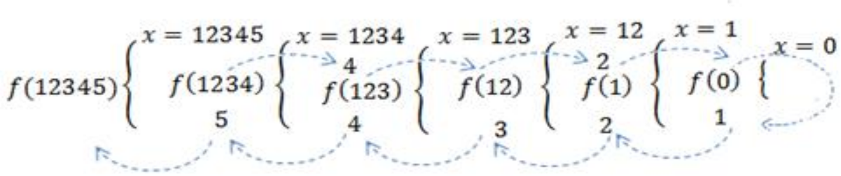
Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Subprogramul **MinMax** primește prin intermediul parametrului **v**, un tablou unidimensional cu maximum 100 de numere reale. Subprogramul mută, la începutul tabloului, toate aparițiile celui mai mic număr din tablou și, la sfârșitul tabloului, toate aparițiile celui mai mare număr din tablou, fără a modifica ordinea celorlalte elemente din tablou. Scrieți definiția completă a subprogramului **MinMax**. **(10p)** **Exemplu**. Pentru valorile **n=10**, **v=(5, 8, 2, 9, 2, 7, 9, 4, 2, 8)**, subprogramul **MinMax** va furniza prin parametrul **v** tabloul: **(2, 2, 2, 5, 8, 7, 4, 8, 9, 9).** **(10p.)**
2.	Un şir de maximum **255** caractere conţine cuvinte scrise cu litere mici. Oricare două cuvinte alăturate sunt separate printr-un spaţiu. Scrieţi programul **Pascal|C/C++** care citește un astfel de șir și înlocuiește fiecare cuvânt ce conține cel puțin trei litere cu un nou cuvânt format cu prima și ultima literă a cuvântului. Șirul astfel obținut se afișează pe ecran. De exemplu, pentru șirul: **`am vazut o raza de soare`** se afișează pe ecran noul șir: **`am vt o ra de se`** **(10p.)**
3.	Numim secvenţă liniară o secvenţă de numere întregi $a_1.a_2,a_3,...,a_n$ cu proprietatea că $a_{i+1} - a_i = i$, (∀) $1 \le i \le n-1$ Fişierul text **numere.in** conţine un şir **s** de cel mult un milion de numere naturale nenule, formate fiecare din cel mult **4** cifre, separate prin câte un spaţiu. **a**) Scrieţi un program Pascal | C/C++ care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină şi afişează pe ecran cea mai lungă secvenţă liniară din şir. Elementele secvenţei liniare se afişează separate prin câte un spaţiu. Dacă există mai multe secvenţe liniare de lungime maximă, se afişează una din ele. **Exemplu:** dacă fişierul **numere.in** are conţinutul: **`4 6 8 9 11 14 5 7 9 10 12 15 19 4 2`** atunci, pe ecran se va afişa secvenţa liniară: **`9 10 12 15 19`**. **(8p.)** **b)** Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

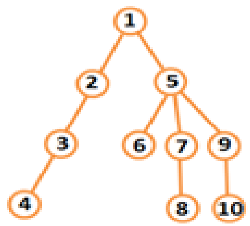
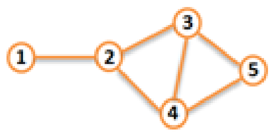
⚠ **Observaţii:**

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 20 - *Rezolvare*

Subiectul 1

<table>
<tr><td>1.</td><td colspan="2">Pentru precizarea răspunsului d) se acordă 4p.
Pentru eliminarea variantelor incorecte se pot considera cazuri particulare ale parității numerelor memorate în variabilelor a și b.</td></tr>
<tr><td>2.</td><td colspan="2">Imaginea următoare reprezintă o reprezentare schematică a modului în care se execută apelul recursiv f(12345).


Pentru răspunsul c) 4212345, se acordă 4p.</td></tr>
<tr><td>3.</td><td colspan="2">Pentru precizarea răspunsului c) se acordă 4p.
Primele 8 submulțimi generate sunt: {1}, {1.2}, {1,2,3}, {1,2,3,4}, {1,2,4, {1,3}, {1,3,4}, {1,4}.</td></tr>
<tr><td>4.</td><td>Pentru precizarea răspunsului c) se acordă 4p.
Nodurile 4, 8 și 10 au câte 3 ascendenți.</td><td>
</td></tr>
<tr><td>5.</td><td>Pentru precizarea răspunsului d) se acordă 4p.
}n graful dat sunt 13 subgrafuri complete: {1}, {2}, {3}, {4}, {5}, {1,2}, {2,3}, {2,4}, {3,4}, {3,5}, {4,5}, {2,3,4}, {3,4,5}.</td><td>
</td></tr>
</table>

Subiectul 2

1.

a) Din cele două numere citite se elimină cifre, începând cu cifra unităților, până când se elimină toate cifrele dintr-un număr sau până când întâlnim cifre diferite.

În numărul memorat de variabila `a`, fiecare cifră eliminată se înlocuiește cu `0`.

Pentru răspunsul `12300` se acordă `6p`.

b) Pentru a putea obține un număr divizibil cu 50, numărul afișat trebuie să fie de forma $\overline{xy00}$ sau $\overline{xy50}$. Valoarea citită în variabila a trebuie să aparțină reuniunii de mulțimi: $\{\overline{xy09} \mid x, y \in \{1,\ldots,8\}, x \neq y\} \cup \{\overline{xy89} \mid x \in \{1,\ldots,7\}, y \in \{0,\ldots,7\}, x \neq y\} \cup$

$\{\overline{xy59} \mid x \in \{1,\ldots,8\} \setminus \{5\}, y \in \{0,\ldots,8\} \setminus \{5\}, x \neq y\} \cup \{\overline{xy50} \mid x, y \in \{1,\ldots,9\} \setminus \{5\}, x \neq y\}$

Cardinalul reuniunii de mulțimi = 8x7+7x7+ 7x7+8x7= 56+49+49+56=210.

Pentru răspunsul `210` se acordă `6p`.

c) Înlocuirea structurii `cât timp…execută`, o structură repetitivă de alt tip, presupune utilizarea unei structuri repetitive `repetă-până când` sau `repetă-cât timp`:

```
citește a,b
p←1
┌ dacă a*b>0 și a%10=b%10 atunci
│┌repetă
││ a←[a/10]
││ b←[b/10]
││ p←p*10
│└până când a*b=0 sau a%10!=b%10
└■
a←a*p
scrie a
```

```
citește a,b
p←1
┌ dacă a*b>0 și a%10=b%10 atunci
│┌repetă
││ a←[a/10]
││ b←[b/10]
││ p←p*10
│└cât timp a*b>0 și a%10=b%10
└■
a←a*p
scrie a
```

Pentru alegerea unei structuri repetitive se acordă 2p., pentru testarea inițială a condiției se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 1p, iar pentru scrierea integrală a restului algoritmului se mai acordă 1p. - în total **6p**

d) Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului se acordă câte 1p., pentru instrucţiunea repetitivă se acordă 4p., iar pentru cele 5 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.

2.

Arcele `a1` și `a2` sunt incidente dacă au o extremitate comună.

Un răspuns corect:

Pascal: `(a1.x=a2.x)or (a1.x=a2.y) or (a1.y=a2.x) or (a1.y==a2.y)`

C/C++: `a1.x==a2.x || a1.x==a2.y || a1.y==a2.x || a1.y==a2.y`

Pentru orice răspuns corect se acordă **6p**.

3.

La finalul executării secvenței,matricea `M` conține elementele din imaginea alăturată. Cea mai mare valoare memorată în matrice este `12`, valoare memorată în elementul aflat pe linia `5`, coloana `4`.

Se acordă câte `2p`. pentru fiecare element identificat corect (valoare maximă, linie, coloană), în total `6p`.

-1	1	0	2	1
3	2	4	3	5
4	6	5	7	6
8	7	9	8	10
9	11	10	12	11

Subiectul 3

1.	Soluția prezentată determină mai întâi valoarea minimă, numărul de apariții ale valorii minime și valoarea maximă. Se utilizează un vector suplimentar în care se adaugă mai întâi valorile minime, apoi, în ordine, elementele diferite de valoarea minimă și de valoarea maximă, și la sfârșit, valorile maxime. Vectorul astfel obținut este copiat în vectorul **v**, obținându-se ordinea dorită.

Limbajul Pascal

```pascal
Type vector=array[0..99]of real;
procedure MinMax(n:byte;var v:vector);
var vmin,vmax:real;
    k,i,kmin:byte;
    w:vector;
begin
 kmin:=1; vmin:=v[0];vmax:=v[0];
 for i:=1 to n-1 do
  if v[i]>vmax then vmax:=v[i]
  else if v[i]=vmin then inc(kmin)
       else if v[i]<vmin then
         begin vmin:=v[i];kmin:=1 end;
 for i:=0 to kmin-1 do w[i]:=vmin;
 k:=kmin;
 for i:=0 to n-1 do
   if(v[i]>vmin) and (v[i]<vmax) then
      begin w[k]:=v[i];inc(k)
      end;
 while k<n do begin w[k]:=vmax;inc(k)
             end;
 for i:=0 to n-1 do v[i]:=w[i]
end;
```

Limbajul C/C++

```cpp
void MinMax(int n, float v[100])
{ int i,kmin=1,k;
  float
vmin=v[0],vmax=v[0],w[100];

  for(i=1;i<n;i++)
    if(v[i]>vmax) vmax=v[i];
    else
      if(v[i]==vmin)kmin++;
      else
      if(v[i]<vmin)
          vmin=v[i],kmin=1;
  for(i=0;i<kmin;i++)w[i]=vmin;
  k=kmin;
  for(i=0;i<n;i++)
   if(v[i]>vmin && v[i]<vmax)
                 w[k++]=v[i];
  while(k<n)w[k++]=vmax;
  for(i=0;i<n;i++)v[i]=w[i];
}
```

Pentru antet corect se acordă **2p** (**1p** – procedure/void; **1p**- parametri intrare).

Pentru determinarea valorii minime, a numărului de apariții ale valorii minime și pentru determinarea valorii maxime, câte **1p.**

Pentru repoziționarea elementelor minime, a elementele maxime și a celorlalte elemente din tablou, câte **1p.**

Pentru declararea variabilelor locale se acordă **1p**, iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă **1p**.

2. Pot fi concepute mai multe metode de rezolvare a problemei.

Prezentăm două soluții posibile, prima soluție (**Pascal**) realizează modificările direct asupra șirului citit, iar cea de-a doua soluție (**C++**) construiește un șir nou ce va memora caracterele șirului obținut în urma prelucrării.
Pentru declararea corectă a variabilelor se acordă **1p**.
Pentru citire a datelor se acordă **1p**. Pentru identificarea cuvintelor ce au cel puțin **3** litere se acordă **1p.**
Pentru înlocuirea cuvintelor ce conțin cel puțin **3** litere cu un nou cuvânt, format cu

prima și ultima literă din cuvânt se acordă **2p**, iar pentru obținerea noului șir (conform cerinței) se acordă **3p**.
Pentru afișarea noului șir în formatul cerut se acordă **2p.**
Pentru corectitudinea globală a programului se acordă **1p** - în total **10p**.

Limbajul Pascal

```
var s:string;
    i,j:integer;
begin
readln(s);
s:=s+' ';
i:=1;
while i<length(s) do
 begin
    j:=i+1;
    while s[j] <> ' ' do inc(j);
    if j-i>2 then begin
      delete(s,i+1,j-i-2);
      i:=i+2
                   end
    else
       while s[i]<>' 'do inc(i);
    inc(i);
 end;
 delete(s,length(s),1);
 write(s);
 end.
```

Limbajul C/C++

```
#include <iostream>
#include <string.h>

using namespace std;
char s[256],snou[256],*p,aux[3]="  ";
int main()
{ cin.getline(s,256);
  p=strtok(s," ");
  while(p)
  { if(strlen(p)>2)
       { aux[0]=p[0];
         aux[1]=p[strlen(p)-1];
         strcat(snou,aux);
       }
    else strcat(snou,p);
    strcat(snou," ");
    p=strtok(NULL," ");
  }
  snou[strlen(snou)-1]=0;
  strcpy(s,snou);
  cout<<s;
  return 0;
}
```

3. **b)** În algoritmul propus am notat cu **l**, lungimea secvenței liniare corecte, cu **lmax**, lungimea celei mai lungi secvențe liniare și cu **u**, ultimul număr din cea mai lungă secvență liniară.

Penru fiecare pereche **(x,y)** de numere citite consecutiv se verifică dacă **x** precede **y** în secvența liniară. În caz afirmativ, se incrementează lungimea secvenței liniare curente, iar altfel, se actualizează valoarea variabilelor **u** și **lmax**, și apoi se resetează lungimea secvenței liniare curente.

Eficiența algoritmului, ca timp de executare, constă în faptul că determinăm valorile **u** și **lmax** (care ne ajută să afișăm secvența cerută) în timpul citirii datelor, deci parcurgem o singură dată șirul de numere. Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple, fără folosirea unor structuri suplimentare de date.

Pentru o descriere coerentă a metodei se acordă **1p.**, iar pentru justificarea eficienței, încă **1p.** - în total **2p**.

	a)	**Limbajul Pascal**	**Limbajul C/C++**

Limbajul Pascal

```
var f:text;
    x,y,u,l,lmax:longint;
begin
 assign(f,'numere.in');reset(f);
 read(f,x); l:=1;u:=x;
 while not eof(f) do begin
   read(f,y);
   if y-x=1 then inc(l)
    else begin
          if l>lmax then begin
             lmax:=l;u:=x end;
          l:=1
         end;
   x:=y
  end;
  if l>lmax then begin
         lmax:=l; u:=y end;
  x:=u-lmax*(lmax-1) div 2;
  for l:=1 to lmax do
    begin write(x,' ');
          x:=x+1
    end;
  close(f);
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>

using namespace std;
ifstream f("numere.in");
int x,y,u,l,lmax;

int main()
{ f>>x;l=1;u=x;
  while(f>>y)
    { if(y-x==1)l++;
       else {if(l>lmax)lmax=l,u=x;
             l=1;
            }
      x=y;
    }
  if(l>lmax) lmax=l,u=y;
  x=u-lmax*(lmax-1)/2;
  for(l=1;l<=lmax;l++)
    { cout<<x<<' ';
      x=x+1;
    }
  f.close();
  return 0;
}
```

Se acordă **1p**. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă **1p**. pentru citirea tuturor numerelor din fişier, **1p.** pentru un algoritm principial corect, încă **1p.** pentru determinarea secvenței liniare de lungime maximă, **1p.** pentru afișarea ei, **1p.** pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă **1p.** pentru alegerea unui algoritm eficient ca timp de executare (**O(n)**) și **1p.** pentru utilizarea eficentă a memoriei. În total, **8p**.

VARIANTA 21

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Știind că variabila **a** memorează un număr natural format din cel puțin două cifre, stabiliți care dintre expresiile **Pascal/C/C++** următoare are valoarea **true/1** dacă şi numai dacă valoarea variabilei **a** are cifra zecilor strict mai mare decât cifra unităților.

	Limbajul Pascal	Limbajul C/C++
a.	`a mod 10 div 10 > a div 10`	`a%10/10 > a/10`
b.	`a div 100 mod 10 > a mod 10`	`a/100%10 > a%10`
c.	`a div 10 mod 10 > a div 10`	`a/10%10 > a/10`
d.	`a div 10 mod 10 > a mod 10`	`a/10%10 > a%10`

2. Pentru subprogramul **f** cu definiția următoare, indicați valoarea care se va afișa la apelul **f(20)**.

Limbajul Pascal

```
function f(x:integer): integer;
begin
  if x=0 then f:=0
  else
    if x mod 3 >0 then f:=f(x-1)
    else f:=x+f(x div 3);
end;
```

Limbajul C/C++

```
int f(int x)
{
 if(x==0) return 0;
 if(x%3>0) return f(x-1);
 return x+f(x/3);
}
```

a. 24	b. 36	c.105	d. 210

3. Utilizând metoda backtracking se generează în ordine lexicografică toate numerele formate din cel puțin trei cifre din mulțimea **{1,2,4,6,7}**, numerele au cifre distincte două câte două și nu conțin două cifre pare alăturate. Primele șase numere generate sunt, în această ordine: **127**, **1274**, **1276**, **147**, **1472**, **1476**. Indicați care sunt ultimele două numere generate, în ordinea generării lor:

a. 761, 7612	b. 67412, 764	c. 6714, 7614	d. 7612, 7614

4. Se consideră arborele cu **10** noduri, etichetate cu numere distincte de la **1** la **10**, dat prin următorul vector de taţi: **(5,5,2,5,0,2,2,9,7,7)**. Stabiliţi care este numărul descendenți ai nodului **2** din arbore:

a.1	b. 3	c. 4	d. 6

5. Se consideră un graf neorientat complet G, cu 2021 noduri, etichetate cu numerele distincte de 1 la 2021. Indicați care este numărul componentelor conexe ale grafului parțial rezultat prin ștergerea tuturor muchiilor incidente cu nodurile 1 și 2 din graf.

a. 1	b. 2	c. 3	d. 4

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte a (număr natural)
d←a%10; b←a; p←1
┌cât timp a>0 execută
│ c←a%10
│ ┌dacă c≥d atunci
│ │ ┌pentru i←1,d execută
│ │ │       b←b-p
│ │ └■
│ └■
│ a←[a/10]; p←p*10
└■
scrie b
```

a) Ce valoare va fi afişată dacă se citesc valorile **15302**? **(6p.)**

b) Scrieţi trei numerele naturale distincte, formate fiecare din câte cinci cifre, care pot fi citite pentru variabila **a** astfel încât, pentru fiecare astfel de valoare, să se afişeze valoarea **1230**. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **pentru...execută** cu o instrucţiune de atribuire. **(6p.)**

d) Scrieţi programul **Pascal/C/C++** corespunzător algoritmului dat. **(10p.)**

2. Se consideră declarările următoare în care variabila **C** memorează informaţiile referitoare la un cerc din planul cu sistemul de coordonate **xOy**: coordonatele centrului cercului şi raza cercului în sistemul de coordonate **xOy**. Scrieţi o expresie în limbajul **Pascal|C/C++** care să aibă valoarea **true/1** dacă şi numai dacă cercul (cu informaţiile memorate în variabila **C**) este situat în cadranul **I**. **(6p.)**

Limbajul Pascal

```
Type punct=record
           x,y:integer
end;
cerc = record
       O:punct; R: integer;
end;
var C: cerc;
```

Limbajul C/C++

```
struct punct {
        int x,y;
};
struct cerc {
       struct punct O; int R;
};
cerc C;
```

3. În secvenţa următoare, variabilele **i** şi **j** sunt de tip integer **inter/int**, iar variabila **A** memorează o matrice cu **5** linii şi **5** coloane (numerotate de la **0** la **4**) cu elemente de tip **integer/int**. Scrieţi cea mai mare şi cea mai mică valoare memorată în matricea **A** la finalul executării secvenţei. **(6p.)**

Limbajul Pascal

```
for i:= 0 to 4 do begin
   for j:= 4 downto 0 do
         A[i,j]:=(i+j)mod 5;
```

Limbajul C/C++

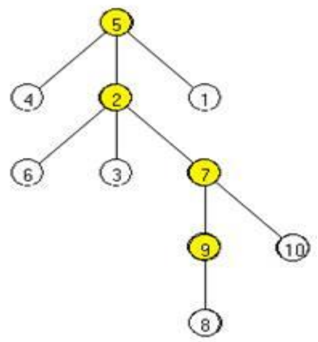
```
for(i=0;i<=4;i++)
   for(j=4;j>=0;j--)
         A[i][j]=(i+j)%5;
```

Subiectul 3 (30p)

	Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.
1.	Scrieţi în caseta text următoare definiţia completă a unui subprogram **sub**, cu doi parametri, care primeşte prin intermediul parametrilor: • **n**, un număr natural nenul (**0 < n < 100**); • **v**, un tablou unidimensional ce memorează un şir de **n** numere naturale, fiecare având cel mult patru cifre. Subprogramul **sub** determină cel mai mic și cel mai mare număr din şirul de numere primit prin parametrul **v** și modifică acest șir prin eliminarea primei apariții a celui mai mic număr din șir și dublarea ultimei apariții a celui mai mare număr din șir, ca în exemplul de mai jos. De asemenea, subprogramul va furniza prin intermediul parametrului **v** tabloul modificat. Dacă șirul nu conține cel puțin două numere distincte, atunci șirul de numere primit prin parametrul **v** nu se va modifica. **Exemplu**. Pentru valorile **n=10**, **v=(35, 98, 52, 4, 98, 51, 1, 11, 1, 65)**, ale parametrilor, în urma apelului, subprogramul **sub** va furniza prin parametrul **v** tabloul: **(35, 98, 52, 4, 98, 98, 51, 11, 1, 65).** **(10p.)**
2.	Un şir cu maximum `255` de caractere conţine cuvinte formate numai din litere mici ale alfabetului englez. Cuvintele din text sunt separate prin câte un spaţiu. Scrieţi un program **Pascal/C/C++** care citeşte un astfel de şir, apoi modifică șirul în memorie prin eliminarea tuturor cuvintelor din şir (cu excepția primului cuvânt) care au același număr de vocale ca și primul cuvânt din text. Șirul rezultat se va afișa pe ecran. ***Exemplu***. Pentru şirul: **`cartea de fizica si cea de romana sunt interesante`** se va afişa: **`cartea de  si cea de  sunt interesante`** **(10p.)**
3.	Fişierul text bac.in conţine un şir **s** de cel mult un milion de numere naturale nenule, formate fiecare din cel puțin **2** și cel mult **9** cifre, separate prin câte un spaţiu. **a**) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină și afișează pe ecran lungimea maximă a unei secvenţe din şir formată doar din numere care au aceeași cifră a zecilor. O secvenţă a unui şir constă în elemente situate pe poziţii consecutive în şirul considerat. **(8p.)** **Exemplu:** dacă fişierul **bac.in** are conţinutul: **`1010 110 10 245 33 32 134 39 51 352 451 3150 11 178 76 81`** atunci, pe ecran se va afişa numărul **4** reprezentând lungimea maximă a unei secvenţe formată doar din numere care au aceeași cifră a zecilor și sunt situate pe poziții consecutive în şirul dat. **b)** Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

VARIANTA 21 - *Rezolvare*

Subiectul 1

<table>
<tr><td>1.</td><td colspan="2">Se accesează cifrele unităților și zecilor numărului memorat în variabila a folosind operatorii <code>div | /</code> și <code>mod | %.</code> Pentru precizarea răspunsului <code>d)</code> se acordă 4p.</td></tr>
<tr><td>2.</td><td colspan="2"><code>f(20)=f(19)=f(18)=18+f(6)=18+6+f(2)=18+6+f(1)=18+6+f(0)=18+6+0=24</code>
Pentru răspunsul a) se acordă 4p.</td></tr>
<tr><td>3.</td><td colspan="2">Se generează în maniera "înaintare cu revenire", cuvintele:
127, 1274, 1276, 147, 1472, 1476, etc. Pentru răspunsul d) se acordă 4p.</td></tr>
<tr><td>4.</td><td>Din vectorul de referințe ascendente deducem că rădăcina arborelui este nodul 5 (tata[5]=0). În imaginea alăturată este reprezentat arborele dat prin vectorul de tați din enunt.
Observăm că descendenții nodului 2 sunt nodurile: 3,6,7,8,9,10.
Pentru precizarea răspunsului d) se acordă 4p.</td><td></td></tr>
<tr><td>5.</td><td colspan="2">Graful parțial obținut va conține 3 componente conexe :
• componenta ce conține doar nodul 1
• componenta ce conține doar nodul 2
• componenta ce conține nodurile 3,4,5,...2021.
Pentru precizarea răspunsului c) se acordă 4p.</td></tr>
</table>

Subiectul 2

<table>
<tr><td rowspan="3">1.</td><td>a)</td><td>Se înlocuiește în numărul a fiecare cifră mai mare decât cifra unităților cu diferența dintre aceste două cifre.
Pentru precizarea valorii 13100 se acordă 6p.</td></tr>
<tr><td>b)</td><td>Pentru variabila a poate fi citit orice număr natural de cinci cifre xyztw care are x≥3, x=w, yzt=123 sau yzt=(1+x)(2+x)(3+x).
De exemplu: 41234, 51235, 61236 sau 45674
Pentru precizarea celor trei valori corecte se acordă 6p. Pentru fiecare valoare corectă se acordă câte 2p.</td></tr>
<tr><td>c)</td><td>Înlocuim structura <code>pentru...execută</code> cu atribuirea <code>b←b-p*d</code>:</td></tr>
</table>

```
citește a (număr natural)
d←a%10; b←a; p←1
┌cât timp a>0 execută
│ c←a%10
│ ┌dacă c≥d atunci
│ │    b←b-p*d
│ └■
│ a←[a/10]; p←p*10
└■
scrie b
```

Pentru înlocuirea structuri repetitive cu o instrucțiune de atribuire se acordă **2p.**, pentru echivalența algoritmilor se acorda acordă **2p.**, iar pentru corectitudinea globală a algoritmului înlocuit se mai acordă 2**p.** - în total **6p**.

d) Pentru instrucţiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și de decizie se acordă câte **1p**., pentru cele două instrucţiuni repetitive se acordă **3p**., iar pentru cele **6** atribuiri se acordă **2p**. Pentru structura corectă a programului se mai acordă **1p.**, în total **10p**.

2. Un răspuns corect: `C.O.x>=C.R && C.O.y>=C.R (C/C++)`, respectiv `C.O.y>=C.R and C.O.y>=C.R (Pascal).`
Pentru orice răspuns corect se acordă **6p**.

3. Răspuns corect: `0 4`
La finalul executării secvenței date conținutul matricei A este:

```
0 1 2 3 4
1 2 3 4 0
2 3 4 0 1
3 4 0 1 2
4 0 1 2 3
```

Se acordă câte 3**p**. pentru fiecare valoare indicată corect.

Subiectul 3

1. Prelucrarea oferită de subprogram determină poziția primei apariții a valorii minime din tablou, poziția ultimei apariții a valorile maxime în șir, apoi elimină cea mai mică, valoare și dublează cea mai mare valoare din șir, situate pe pozițiile determinate. Tabloul rezultat este returnat prin intermediul parametrului **v** al subprogamului.

Limbajul Pascal

```
Type vector=array[1..100]of integer;
procedure sub(n:byte;var v:vector);
var pmin,pmax,i:byte;
    aux:integer;
begin
        pmin:=1;
        pmax:=1;
```

```
        for i:=1 to n do
                if v[i]>=v[pmax] then
                        pmax:=i
                else
                    if v[i]<v[pmin]
                        then pmin:=i;
        if v[pmin]<>v[pmax] then
           if pmin < pmax then
               for i:= pmin+1 to pmax do begin
                  v[i-1]:=v[i]; end
           else
               for i:= pmin downto pmax+1 do begin
                  v[i]:=v[i-1]; end;
end;
```

Limbajul C/C++

```
void sub(int n, int v[])
{
    int i, pmin=1, pmax=1,aux;
    for(i=2; i<=n; i++)
        if (v[i]>=v[pmax])
           pmax=i;
        else
        if (v[i]<v[pmin])
           pmin=i;
    if (v[pmin]!=v[pmax])
        if(pmin<pmax)
          for(i=pmin+1; i<=pmax; i++)
            v[i-1]=v[i];
        else
          for(i=pmin; i>=pmax+1; i--)
            v[i]=v[i-1];
}
```

Pentru antet corect se acordă **2p**. Pentru determinarea numărului minim din tabloul **v** se acordă **1p**, iar pentru determinarea poziției primei apariții a minimului se acordă **2p**. Pentru determinarea numărului maxim din tablou **v** se acordă **1p**, iar pentru determinarea poziției ultimei apariții a maximului se acordă **1p**. Pentru eliminarea primului minim se acordă **1p**. Pentru dublarea ultimului maxim se acorda **1**p. Pentru corectitudinea globală a programului **1p**.

2. O soluție se poate obține separând cuvintele din text, numărând vocalele fiecărui cuvânt și comparându-le cu numărul nrv de vocale ale primului cuvânt din text. Cuvintele ce au nrv vocale vor fi eliminate din text cu excepția primului cuvânt. Se poate construi un șir nou care să conțină cuvintele ce vor rămâne în text, prin concatenarea acestora și inserarea caracterului spațiu după fiecare cuvânt.
Se pot utiliza funcţiilor predefinite pentru şiruri de caractere specifice limbajului.
Pentru declararea corectă a variabilelor se acordă **1p**. Pentru citirea șirului se

acordă **1p**. Pentru separararea primului cuvânt din text se acordă **1p.** Pentru determinarea numărului de vocale nrv se acordă **1p.** Pentru determinarea unui cuvânt din text care are nrv vocale și pentru eliminarea lui se acordă **2p**. Pentru eliminarea tuturor cuvintelor din șir cu proprietatea cerută se acordă **2p**. Pentru modificarea și afișarea șirului în formatul cerut se acordă 1**p.** Pentru corectitudinea globală a programului se acordă **1p** - în total **10p**.

Limbajul Pascal	Limbajul C/C++
<pre>var i,n,k,nrv,nr :byte; s,p,q :string; begin readln(s); s:=s+' '; n:=pos(' ',s); for i:=1 to n-1 do if pos(s[i],'aeiou')>0 then inc(nrv); p:=copy(s,1,n); delete(s,1,n); k:=pos(' ',s); while k>0 do begin p:=p+' '; nr:=0; for i:=1 to k-1 do if pos(s[i],'aeiou')>0 then inc(nr); if nrv<>nr then begin q:=copy(s,1,k); p:=p+q end; delete(s,1,k); k:=pos(' ',s); end; s:=p; writeln(s); end.</pre>	<pre>int main() { char s[256], t[256], *p, *q; int i,nrv=0,nr; cin.getline(s,256); p=strtok(s," "); strcpy(t,p); strcat(t," "); for(i=0; i<strlen(p); i++) if(strchr("aeiou",p[i])) nrv++; q=strtok(NULL," "); while(q) { nr=0; for(i=0; i<strlen(q); i++) if(strchr("aeiou",q[i])) nr++; if(nr!=nrv) strcat(t,q); strcat(t," "); q=strtok(NULL," "); } strcpy(s,t); cout<<s; return 0; }</pre>

3. **b)** Algoritmul de rezolvare propus determină cifra zecilor primului număr c**x** și apoi citește un număr din fișier, comparând cifra zecilor acestuia cu c**x**: în caz de egalitate se "prelungește" secvența curentă, în caz contrar, se trece la o nouă secvență cu o nouă cifra c**x** curentă. La finalul oricărei secvențe se analizează lungimea ei, în vederea memorării lungimii maxime.
Eficiența algoritmului, ca timp de executare, constă în faptul că determinăm valoarea cerută în timpul citirii datelor, deci parcurgem o singură dată șirul de numere. Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple, fără folosirea unor structuri suplimentare de date.

Pentru o descriere coerentă a metodei se acordă **1p.**, iar pentru justificarea eficienței, încă **1p**. - în total **2p**.

a)

Limbajul Pascal

```
var f:text;
        nr,lgmax,lg,x,cx,snr:longint;
begin
        assign(f,'bac.in');reset(f);
        read(f,x);
        lg:=1; cx:=x div 10 mod 10;
        while not(eof(f)) do begin
                read(f,nr);
                cnr:=nr div 10 nr mod 10;
                if cnr=cx then inc(lg)
                else begin
                        if lgmax<lg then lgmax:=lg;
                        lg:=1; cx:=cnr
                end;
        end;
        if lgmax<lg then lgmax:=lg;
        close(f);
        write(lgmax)
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;

ifstream f("bac.in");
int main()
{       int lgmax=0, lg=1, nr, cx=0, x, cnr;
        f>>x;
        cx=x/10%10;
        while(f>>nr)
        {       cnr=nr/10%10;
                if (cnr==cx) lg++;
                else {  if(lgmax<lg) lgmax=lg;
                        lg=1; cx=cnr;
                }
        }
        if (lgmax<lg) lgmax=lg;
        cout<<lgmax;
        return 0;
}
```

Se acordă **1p**. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă **1p**. pentru citirea tuturor numerelor din fişier, **1p.** pentru un algoritm principial corect, încă **1p.** pentru determinarea valorii cerute, **1p.** pentru afișarea ei, **1p.** pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.). Se acordă **1p.** pentru alegerea unui algoritm eficient ca timp de executare (**O(n)**) și **1p.** pentru utilizarea eficentă a memoriei (**O(1)**).
În total, **8p**.

VARIANTA 22

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Indicați ce valori va avea variabila de tip întreg **x** după ce se execută următoarea secvență de operații.

Limbajul Pascal

```
x:=2021 div 100;
y:=123 mod 10*3 div 8;
x:=x+y;
write(x);
```

Limbajul C/C++

```
x=2021/100;
y=123%10*3/8;
x=x+y;
cout<<x; | printf("%d",x);
```

a. `4`	b. `20`	c. `21`	d. `22`

2. Pentru subprogramul **f** cu definiția următoare, indicați ce se va afișa în urma apelului **f(2)**.

Limbajul Pascal

```
procedure f (n:integer);
var i:integer;
begin
if(n>0) then
  for i:=1 to n do
   begin
    f(n-2);
    write(i,' ')
   end
end;
```

Limbajul C/C++

```
void f(int n)
{int i;
if(n>0)
 for(i=1;i<=n;i++)
  {f(n-2);
  cout<<i<<’ ’;
        | printf(“%d”,i);
   }
}
```

a. `1 0`	b. `1 2`	c. `2 0`	d. `2 1`

3. Utilizând *metoda Backtracking* se generează toate tripletele cu litere din mulțimea `{a,b,c,d,e,f}`, astfel încât să nu fie două vocale alăturate. Primele triplete generate sunt: **`aba, abb, abc, abd, abf,...`** Folosind acceași metodă se generează toate tripletele cu litere din același interval, dar să nu conțină două consoane alăturate. Indicați care sunt ultimele trei triplete generate.

a. `fec fed fef`	b. `fee fec fef`
c. `fee fed fef`	d. `fed fee fef`

4. Se consideră un arbore cu rădăcină care are 2021 de noduri. Indicați numărul minim de frunze pe care îl poate avea arborele.

a. `1`	b. `2`	c. `2020`	d. `2021`

5.	Un graf complet are 5050 muchii. Indicați numărul de noduri ale grafului.			
	a. 50	**b.** 51	**c.** 100	**d.** 101

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Algoritmul următor este reprezentat în pseudocod.**

S-a notat cu **a%b** restul împărţirii numărului întreg **a** la numărul întreg nenul **b** şi cu **[c]** partea întreagă a numărului real **c**.

```
citeşte m,n (numere naturale nenule)
a← m
b← n
r← a%b
┌cât timp r<>0 execută
| a← b
| b← r
| r← a%b
└■
 a← m/b
 b← n/b
 nr← 0
┌cât timp a<>1 și b<>1 execută
| ┌dacă a>b atunci a← a%b
| | altfel b← b%a
| └■
| nr← nr+1
└■
scrie nr
```

a) Scrieţi ce valoare va fi afişată dacă se citesc valorile **195** şi **120**. **(6p.)**

b) Pentru **n=4**, scrieţi o valoare care trebuie să fie introdusă pentru **m** astfel încât rezultatul afişat să fie **0**. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască prima structură **cât timp...execută** cu o structură repetitivă de alt tip.**(6p.)**

d) Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2. Se consideră următoarele declarări, care memorează date despre un album cu melodii. Scrieți o secvență de instrucțiuni care afișează pentru primele trei melodii din album: titlul, durata și autorul. **(6p.)**

	Limbajul Pascal	Limbajul C/C++
	`type melodie=record` `  titlu:string;` `  durata:real;` `  autor:string; end;` `type album=record;` `  nume:string;` `  m:array[1..10] of melodie;` `end;` `var a:album;`	`struct melodie` `{char titlu[50];` ` float durata;` ` char autor[50];};` `struct album` `{char nume[50];` ` melodie m[10];}` ` a;`
3.	Scrieți ce se afișează pe ecran în urma executării următoarei secvențe, știind că `a=”informatician”` și `b=”informatica”`. **(6p.)**	
	Limbajul Pascal	**Limbajul C/C++**
	`i:=0;` `while(a[i]=b[i])` `  begin` `   delete(a,i,1);` `   delete(b,i,1);` `  end;` `write(a,' ',b);`	`i=0;` `while(a[i]==b[i])` ` {strcpy(a+i,a+i+1);` `  strcpy(b+i,b+i+1);` `  }` `cout<<a<<' '<<b;` `| printf("%s %s",a,b);`

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Scrieţi definiţia completă a subprogramului **sub**, cu doi parametri, care primeşte prin intermediul parametrului **x** un număr natural de cel mult **8** cifre și furnizează prin al doilea parametru **n**, numărul factorilor primi distincți din descompunerea lui **x**. **Exemplu:** pentru **x=60** valoarea returnată prin **n** va fi **3**. În descompunerea numărului **60** sunt trei factori primi: **2, 3,** și **5 (60=2*2*3*5).(10p.)**
2.	Să se construiască o matrice cu n linii şi n coloane ale cărei elemente au valori după cum urmează: elementele aflate pe diagonalele matricei vor avea valoarea **-1**, elementele de pe contur, cu excepţia celor de pe diagonale, vor fi egale cu valoarea absolută a diferenţei indicilor specifici fiecărui element, iar celelalte elementele vor avea o valoare ce reprezintă suma celor trei elemente aflate pe linia de deasupra elementului respectiv: cel aflat pe aceeaşi coloană, cel aflat pe coloana precedentă şi cel aflat pe coloana următoare. De exemplu, elementul de pe contur, aflat pe linia a doua și coloana a cincea va fi egal cu **|2-5|**, iar elementul din interior aflat pe linia a doua și coloana a treia va fi egal cu suma celor trei elemente aflate deasupra sa **(1+2+3)**. Scrieţi programul care citeşte de la tastatură o valoare naturală **n (2≤n≤24)**, construieşte matricea conform cerinţei şi o afişează pe ecran, pe linii, cu spaţii între elementele de pe fiecare linie. **(10p.)**

Exemplu: dacă **n=5**, se afişează matricea:

```
-1   1   2   3  -1
 1  -1   6  -1   3
 2   6  -1   8   2
 3  -1  13  -1   1
-1   3   2   1  -1
```

3. Fişierul de intrare **bac.in** conţine **n** **(2≤n≤100000)** numere naturale, de cel mult **8** cifre, separate printr-un spaţiu liber.

a) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină şi afişează pe ecran în ordine strict descrescătoare, separate printr-un spaţiu, toate cifrele numerelor naturale din fişierul **bac.in** .**(8p.)**

Exemplu: dacă fişierul **bac.in** conţine valorile

123412 87996540 23 66543 10098 1 76 , atunci se afişează:

9 9 9 8 8 7 7 6 6 6 6 5 5 4 4 4 3 3 3 2 2 2 1 1 1 1 0 0 0

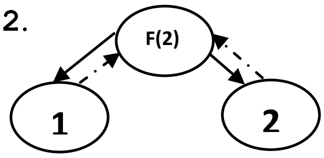
b) Descrieţi în limbaj natural algoritmul proiectat, justificând eficienţa acestuia. **(2p).**

Observaţii:

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 22 - *Rezolvare*

Subiectul 1

1.	Secvența de operații reprezintă o structură linіră, unde ținându-se cont de prioritatea și asocierea operatorilor aritmetici se obține **`x=21`**. Pentru precizarea răspunsului **c)** se acordă **4p**.
2.	Se urmăresc apelurile recursive, pentru **`n=2`**, se afișează **`1 2`**.  Pentru precizarea răspunsului **b)** se acordă **4p**.
3.	Se generează în maniera ”înaintare cu revenire”, tripletele: **`aaa, aab, aac, ...fec, fed, fee, fef.`** Pentru precizarea răspunsului **d)** se acordă **4p**.
4.	Fiecare nod al arborelui are un singur fiu. Cel de al 2021 – lea nod este frunză. Pentru precizarea răspunsului **a)** se acordă **4p**.
5.	Graful complet cu **`n`** noduri are **`n*(n-1)/2`** muchii. Pentru graful cu **`5050`** muchii avem relația **`n*(n-1)=5050*2`**. Așadar graful are **`n=101`** noduri. Pentru precizarea răspunsului **d)** se acordă **4p**.

Subiectul 2

1.	**a)**	Se determină cel mai mare divizor comun al numerelor **`m`** și **`n`** prin intermediul algoritmului lui Euclid prin împărțiri repetate. Cele două numere **`a`** și **`b`** reprezintă numărătorul respectiv numitorul unei fracții ireductibile obținută prin împărțiri repetate ale lui **`m`** respectiv **`n`** la cel mai mare divizor comun al lor. Pentru **`m=195`** și **`n=120`** se obține **`a=13`** și **`b=8`** deoarece cel mai mare divizor comun al numerelor **`m`** și **`n`** este **`15`**, iar **`195/15=13`** și **`120/15=8`**. Variabila **`nr`** numără de câte ori are loc transformarea fracției ireductibile **`13/8`** până când numărătorul sau numitorul acesteia devine **`1`**. Transformarea fracției se realizează astfel: se calculează maximul respectiv minimul dintre **`a`** și **`b`** și se înlocuiește maximul cu restul împărțirii valorii maxime la valoarea minimă. Variabila **`nr`** va avea valoarea **`4`** deoarece fracția ireductibilă **`13/8`** se transformă de patru ori astfel: **`5/8, 5/3, 2/3, 2/1`**. Pentru precizarea valorii **`4`** se acordă **6p**.
	b)	Valoarea variabilei **`nr`** este **`0`** când cea de a doua structură repetitivă **cât timp ...** nu se execută. În acest caz **`a=1`** sau **`b=1`**. Cum **`a=m/cmmdc, b=n/cmmdc`** și **`n=4`** valoarea variabilei **`m`** este un divizor al lui **`n`** sau este egală cu un multiplu al valorii lui **`n`**. Pentru precizarea valorii **`1`** sau **`2`** sau **`4`** sau **`8`** sau **orice multiplu al valorii `4`**, se acordă **6p**.
	c)	Structura **cât timp... execută** poate fi înlocuită cu structura repetitivă **repetă - până când...**

```
a← m
b← n
┌repetă
| r← a%b
| a← b
| b← r
└■ până când r=0
a← m/a
b← n/a
```

Pentru alegerea unei structuri repetitive se acordă 2 p., pentru testarea condiției de oprire acordă 2p și pentru rescrierea integrală a algoritmului se acordă 2p. – în total **6p.**

d) Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc.
O posibilă rezolvare este următoarea:

Limbajul Pascal

```
var n,m,a,b,r,nr:integer;
BEGIN
     nr:=0;
     read(m,n);
a:=m;b:=n;r:=a mod b;while(r<>0)do
     begin
       a:=b;b:=r;r:=a mod b;
     end;
     a:=n div b;b:=m div b;
     while(a>1) and (b>1) do
      begin
       if(a>b)then a:=a mod b
       else b:=b mod a;
       nr:=nr+1;
      end;
write('nr=',nr);
END.
```

Limbajul C/C++

```
#include <iostream>  | #include <cstdio>
using namespace std;
int main()
{long m,n,a,b,r;
 int nr=0;
 cin>>m>>n;| scanf("%ld %ld",&m,&n);
 a=m;b=n;r=a%b;
```

```
 while(r)
  {a=b;b=r;r=a%b;}
  a=m/b; b=n/b;
 while(a!=1 && b!=1)
  {if (a>b) a=a%b;
    else b=b%a;
   nr++;}
 cout<<nr; | printf("%ld",nr);
 return 0;
}
```

Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și de decizie se acordă câte 1p., pentru prima instrucţiune repetitivă se acordă 3p., pentru a doua instrucţiune repetitivă se acordă 3p., iar pentru atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.

2. Un exemplu de implementare:

Limbajul Pascal

```
for i=1 to 3 do
begin
write(a.m[i].titlu,' ',a.m[i].durata,' ',a.m[i].autor);
writeln;
end;
```

Limbajul C/C++

```
 for(i=1;i<=3;i++)
{cout<<a.m[i].titlu<;' '<<a.m[i].durata<<' '
<<a.m[i].autor;
cout<<endl;}
|printf("%s%d%s",a.m[i].titlu,a.m[i].durata,a.m[i].autor)
```

Pentru scrierea corectă a structurii repetitive şi afişarea conţinutului variabilei **m,** conform cerinţei, se acordă **(6p.)**

3. Cât timp cele două șiruri au primul caracter identic secvența realizează eliminarea, din ambele șiruri, a caracterului respectiv sau a succesiunii de caractere identice. După eliminare subșirul din dreapta caracterului șters se deplasează spre stânga cu o poziție. În final șirul **a="ian"**, iar **b="a"**.

Pentru precizarea valorilor finale se acordă 3p+3p. – în total **6p**.

Subiectul 3

1. Prelucrarea oferită de subprogram construiește valoarea cerută prin contorizarea factorilor ireductibil distincți, care se obțin când se realizează descompunerea în factori primi a numărului **x**.

	Limbajul Pascal
	`` procedure sub(x:longint; var n:integer); var d,k:integer; begin d:=2;n:=0; while(x>1) do begin k:=0; while(x mod d = 0)do begin x:=x div d; inc(k); end; if(k<>0) then inc(n); inc(d); end; end; ```

```
procedure sub(x:longint; var n:integer);
var d,k:integer;
begin
  d:=2;n:=0;
  while(x>1) do
   begin
    k:=0;
    while(x mod d = 0)do
     begin
         x:=x div d; inc(k);
     end;
    if(k<>0) then inc(n);
    inc(d);
   end;
end;
```

Limbajul C/C++

```
void sub(long x, int &n)
{int d=2;n=0;
  while(x>1)
   {int k=0;
     while(x%d==0)
      {x=x/d; k++;}
      if(k)n++;
      d++;}
}
```

Pentru respectarea structurii antetului (procedure/void) se acordă 1p., pentru declararea corectă a parametrului de intare **x** 1p., pentru declararea corectă a parametrului de ieșire **n** 1p., pentru declararea variabilelor locale 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p. Corectitudinea algoritmică a prelucrării în vederea obținerii valorii cerute este notată cu 4p. Pentru transmiterea naturală a rezultatului prin parametrul **n** se acordă 1p, aceasta însemnând că valoarea lui **n** nu se afișează și nu se returnează explicit. În total **10p.**

2. O posibilă rezolvare pentru construcția matricei cerute este:

Limbajul Pascal

```
var n,j,i: integer;
a:array[1..25,1..25] of integer;
BEGIN
read(n);
for i:=1 to n do
 for j:=1 to n do
 begin
 if (i=j) or (i+j=n+1) then a[i][j]:=-1
  else
```

```
   begin
    if (i=1) or (j=1) or (i=n) or (j=n) then
      begin if(i>j) then a[i][j]:=i-j
       else a[i][j]:=j-i;
      end
    else
     a[i][j]:=a[i-1][j-1]+a[i-1][j]+a[i-1][j+1];
   end;
  end;
for i:=1 to n do
  begin
   for j:=1 to n do
    write(a[i][j]:4,' ');
    writeln;
   end;
END.
```

Limbajul C/C++

```
#include <iostream> | #include <cstdio>
#include <iomanip>
using namespace std;
int main()
{unsigned int n,i,j;
int a[25][25];
cin>>n; | scanf("%d",&n);
for(i=1;i<=n;i++)
 for(j=1;j<=n;j++)
 if((i==j)||(i+j==n+1)) a[i][j]=-1;
   else
    if((i==1||j==1||i==n||j==n)&&((i!=j)||(j+i!=n+1)))
       if(i>j) a[i][j]=i-j;
         else a[i][j]=j-i;
     else a[i][j]=a[i-1][j-1]+a[i-1][j]+a[i-1][j+1];
for(i=1;i<=n;i++)
  {for(j=1;j<=n;j++)
   cout<<setw(4)<<a[i][j]<<' ';
   | printf("%d\t", a[i][j]);
   cout<<endl; | printf("\n");
  }
return 0;
}
```

Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui **n** se acordă 1p., pentru completarea corectă a elementelor de pe diagonale se acordă 1p., pentru completarea corectă a elementelor de pe contur se acordă 2p., pentru completarea corectă a celorlalte elemente se acordă 2p., iar pentru afișarea matricei conform cerinței se acordă 2p. Pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se acordă 1p. – în total **10p**.

3.	**a)** O posibilă rezolvare este următoarea:
	Limbajul Pascal

```
var n:longint; i:byte; f:text;
v:array[0..9] of longint;
BEGIN
assign(f,'bac.in');
reset(f);
while not eof(f) do
 begin
 read(f,n);
 write(n,' ');
 while(n<>0) do
 begin
  inc(v[n mod 10]);
  n:=n div 10; end;
 end;
for i:=9 downto 0 do
  for j:=1 to v[i] do
  write(i,' ');
  close(f);
END.
```

Limbajul C/C++

```
#include <iostream> | #include <cstdio>
#include <fstream>
using namespace std;
ifstream f("bac.in");
int main()
{int i, v[10];
  long n;
  while (f>>n)
   while(n)
    {v[n%10]++;
     n=n/10;}
for(i=9;i>=0;i--)
 for(j=1;j<=v[i];j++)
 {cout<<i<<' '; | printf("%d\t",i);
  v[i]--;}
f.close();
return 0;
}
```

Se acordă 1p. - pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), 1p. - pentru citirea tuturor numerelor din fişier, 1p. - pentru un algoritm principial corect, 1p. - pentru determinarea tuturor valorilor cerute, 1p. - pentru afișarea lor în ordine strict descrescătoare și 1p. - pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor etc.).

Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare (O(n)) și 1p. pentru utilizarea eficentă a memoriei. În total **8p.**

	b) Parcurgem fișierul `bac.in` și reținem într-un vector caracteristic cifrele fiecărui număr astfel: `v[0]` va reţine numărul cifrelor de `0` întâlnite în toate numerele, `v[1]` va reţine toate cifrele de `1` întâlnite în toate numerele etc. La final parcurgem vectorul și afișăm toate cifrele în ordine descrescătoare folosind o structură repetivită cu număr cunoscut de pași și cu contor descrescător. Algoritmul este eficient, ca timp de execuție, deoarece șirul cu foarte multe numere este parcurs o singură dată. Dependența timpului de execuție de dimensiunea problemei este liniară deoarece numărul de valori din fișier este `n`. Din punctul de vedere al spațiului de memorie soluția propusă este eficientă deoarece sunt utilizate variabile simple de lucru și doar un vector cu `10` componente, evitând astfel memorarea tuturor numerelor din fișier. Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. – în total **2p.**

VARIANTA 23

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Ştiind că variabila întreagă **n** memorează un număr natural cu cel puţin 3 cifre, care dintre următoarele expresii **Pascal | C/C++** are ca valoare numărul obţinut după eliminarea ultimelor 2 cifre ale lui **n**?

	Limbajul Pascal	Limbajul C/C++
a.	`n mod 10 div 10`	`n%10/10`
b.	`n div 10 mod 10`	`n/10%10`
c.	`n mod 10`	`n%10`
d.	`n div 100`	`n/100`

2. Pentru subprogramul **`suma`** cu definiția următoare, indicați ce se va afișa în urma apelului **suma(8745)**.

Limbajul Pascal

```
procedure suma( x:longint);
begin
write(x mod 100);
  if x>99 then
      begin
            suma(x div 10);
            write(2* (x mod 10));
      end;
end;
```

Limbajul C/C++

```
void suma(int x)
{
cout<<x%100; /
printf("%d",x%100);
if(x>99) {
    suma(c/10);
cout<<2*(x%10);
    printf("%d",2*(x%10));
  }

}
```

a. 451088740	b. 451074885	c.10877445	d. 457487810

3. Utilizând *metoda backtracking* se generează în ordine crescătoare toate numerele naturale de câte patru cifre în care numerele de pe poziție consecutivă sunt de paritate diferită. Primele cinci numere generate sunt, în ordine: `1010, 1012, 1014,1016, 1018.` Care este cel de-al 6-le și al 7-le număr generat?

a. 1031 si 1032	b. 1030 si 1032	c. 1032 si 1034	d. 1030 si 1034

4. Câte muchii trebuie eliminate dintr-un graf complet cu 20 de noduri pentru ca graful obținut să fie arbore?

a. 19	**b.** 190	**c.** 171	**d.** 181

5.	Se consideră un arbore cu rădăcină cu **17** noduri, etichetate de la **1** la **17**, dat prin vectorul de taţi **4 1 1 0 4 4 6 11 11 11 6 6 5 5 16 1 16** . Stabiliţi care sunt descendenţii nodului 6.			
	a. 1 5 6	**b.** 8 9 10	**c.** 7 11 12	**d.** 7 8 9 10 11 12

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte n (număr natural)
p1<-1; p2<-1
┌cât timp n>0 execută
| c<-n%10; n<-[n/10]
|┌dacă c%2=0 atunci
|| p1<-p1*c
||altfel
|| p2<-p2*c
|└■
└■
scrie p1,' ',p2
```

a) Ce valoare va fi afişată dacă se citeşte pentru **n** valoarea **2339768**? **(6p.)**

b) Scrieţi cea mai mică valoare de 5 cifre disticte care se poate citi pentru **n** astfel încât valoarea lui **p1** sa fie zero. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **cât timp...execută** cu o structură repetitivă de alt tip. **(6p.)**

d) Scrieţi programul **Pascal/C/C++** corespunzător algoritmului dat. **(10p.)**

2. Fie declararea următoare:

Limbajul Pascal	Limbajul C/C++
`type data=record` `zi,luna,an:integer` `end;` `type masina=record` `marca:string;` `pret:integer;` `data_fabricatiei:data` `end;` `var x,y:masina;`	`struct masina` `{` `char marca[20];` `int pret;` `struct {` `int zi,luna, an;}` `data_fabricatie;` `} x,y;`

Scrieţi o expresie în limbajul **Pascal | C/C++** care să afiseze dintre cele **2** variabile **x** şi **y**, marca maşinii cu anul de fabricaţie cel mai nou. **(6p.)**

3.	În secvenţa de program alăturată, variabila **s** memorează un şir cu cel mult 3**0** de caractere. Care este valoare șirului **s** în urma execuției următoarei secvențe de program. **(6p.)**	
	Limbajul Pascal	**Limbajul C/C++**
	`s:='supercalifragilistic';` `for i:=1 to length(s) do` `  if pos(s[i],'aeiou')>0 then` `        s[i]=s[i]-32;` `write(s);`	`strcpy(s,"supercalifragilistic");` `  for(i=0;i<strlen(s);i++)` `  if(strchr("aeiou",s[i])!=0)` `      s[i]=s[i]-32;` `cout<<s;`

Subiectul 3 (30p)

Scrieţi pe foScaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Scrieți definiția completă a unui subprogram **formare** care primește ca parametru un număr natural a de cel mult **9** cifre și returnează prin intermediul parametrului **b** cel mai mare numar natural care se poate forma cu cifre distincte ale numarului **a**. *De exemplu*, dacă pentru a se primește valoarea **6269532** atunci valoare lui **b** va fi **96532**. **(10p.)**
2.	Se cieşte de la tastatură un număr natural **x** de maxim **9** cifre. Să se construiască o matrice pătratică în care numărul de linii și de coloane este egal cu numărul de cifre ale lui **n**, iar elementele matricei sunt in ordine de la ultima coloană la prima coloană cifrele numărului **x**. *De exemplu*, dacă se citeşte de la tastatură **x=305467**, se va construi şi se va afişa pe ecran matricea: 7 6 4 5 0 3 7 6 4 5 0 3 7 6 4 5 0 3 7 6 4 5 0 3 7 6 4 5 0 3 7 6 4 5 0 3 **(10p.)**
3.	Fişierul text **bac.in** conţine un şir **s** de cel mult un milion de numere naturale nenule, formate fiecare din cel mult **9** cifre, separate prin câte un spaţiu. **a**) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină și afișează in fisierul **bac.out**, prima cifră din scrierea numerelor, care apare de cele mai multe ori precum și număr maxim de apariți. Dacă există mai multe cifre cu număr maxim de aparițîi se vor afișa toate separate cu câte un spațiu. **(8p.)** *Exemplu***:** dacă fişierul **bac.in** are conţinutul:

	`243 65 2453 763 789 67 2453 787 987 545 71465 20007` atunci, pe ecran se va afişa 2 7 4 (cifrele 2 și 7 apar de număr maxim de ori, adică de 4 ori) **b)** Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

Observaţii:

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 23 - *Rezolvare*

Subiectul 1

1.	Pentru precizarea răspunsului **d)** se acordă **4p**.
2.	Pentru răspunsul **d)** se acordă **4p**.
3.	Pentru precizarea răspunsului **b)** se acordă **4p**.
4.	Pentru răspunsul **c)** se acordă **4p.**
5.	Pentru precizarea răspunsului **d)** se acordă **4p**.

Subiectul 2

1.	**a)**	Algoritmul calculează în **P1** produsul cifrelor pare și în **P2** produsul cifrelor impare. Se afișează **96** si **567** și se acordă **6p**. Pentru un singur răspuns corect ce acordă **3p.**
	b)	Cea mai mică valoare este **10234**. Se acordă **6p.**
	c)	Înlocuirea structurii **`cât timp n>0 execută...`** presupune utilizarea unei structuri repetitive **execută - cât** timp sau **repetă - până când**:

```
citeşte n (număr natural)
p1<-1; p2<-1
┌daca n>0 atunci
|   ┌ execută
|   | c<-n%10; n<-[n/10]
|   |┌dacă c%2=0 atunci
|   || p1<-p1*c
|   ||altfel
|   || p2<-p2*c
|   |└■
|   └■ cât timp n>0
└■

scrie p1,' ',p2
```

Pentru alegerea unei structuri repetitive se acordă 2p., pentru testarea inițială a valorii lui **n** se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 1p, iar pentru scrierea integrală a restului algoritmului se mai acordă 1p. - în total **6p**.

	d)	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind

		structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc. Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și de decizie se acordă câte 1p., pentru instrucţiunea repetitivă se acordă 3p., iar pentru cele 6 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.
2.	Un răspuns corect: **Pascal:** `if x.data_fabricatiei.an>y.data_fabricatiei.an then` `write(x.marca)` `else` `write(y.marca);` **C/C++:** `if (x.data_fabricatiei.an>y.data_fabricatiei.an)` `cout<<x.marca;` `else` `cout<<y.marca;` **Pentru orice răspuns corect se acordă 6p.**	
3.	Secvența de program transformă vocalele din litere mici în litere mari. Se va afișa: sUpErcAlIfrAgIlIstIc Se acordă **6p.**	

Subiectul 3

1.

Limbajul Pascal

```
procedure formare(a:longint, var b:longint)
var
    i,c,ok:longint;
begin
    b:=0;
    for i:=9 downto 0 do
    begin
        c:=a;
        ok:=0;
        while c<>0 do
        begin
            if c%10=i then
                ok:=1;
            c:=c div 10;
        end;
```

	`` if ok=1 then b:=b*10+i; end; end; ``` **Limbajul C/C++** ``` void formare(long int a, long int &b) { long int i,c,ok; b=0; for (i=9;i>=0;i--) { c=a; ok=0; while (c!=0) { if (c%10==i) ok=1; c=c/10; } if (ok==1) b=b*10+i; } } ``` Pentru antet corect se acordă 3p. (Pentru respectarea structurii antetului (procedure/void) se acordă 1p., pentru declararea corectă a parametrilor de intare a și b încă 1p.). Pentru declararea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p. Pentru determinarea aparitiei unei cifre in a 1p. Pentru formarea numărului 3p. Pentru corectitudinea globală a programului 1p. În total **10p**.
2.	Pot fi concepute mai multe metode de obținere a matricei cerute. Se determină numărul de cifre ale numărului **x,** apoi se completează coloanele matricei de la stânga la dreapta cu cifrele numărului. ``` n<-0 c<-x cat timp c<>0 executa | n<-n+1 | c<-c div 10 └■ Pentru j=1,n executa | Pentru i=1,n executa |	a[i,j]=x mod 10 | └■ | x<-x div 10 └■ ```

	Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui **x** se mai acordă 1p., pentru aflarea numarului de linii si coloane se acorda 1p., pentru completarea corectă a unei coloane se acordă 2p., pentru completarea corectă a tuturor coloanelor se acordă 2p., iar pentru afișarea matricei conform cerinței se acordă 2p. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă 1p. - în total **10p**.

3. **b)** **Algoritmul de rezolvare propus** citește, numerele din fișier, si creează un vector de frecvență din prima cifră a fiecărui număr. Calculăm valoarea maximă din vector și apoi afișăm toate cifrele care apar de număr maxim de ori.

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul. Deoarece numărul de valori din fișier este mare, spunem că am obținut un algoritm liniar. Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar un vector de maxim **10** elemente.

Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. - în total **2p.**

a) **Limbajul Pascal**

```
var f,g:text;
      x,maxi,i:longint;
     a:array[0.11] of integer;
begin
      assign(f,'bac.in');reset(f);
      assign(g,'bac.out');rewrite(f);
      while not eof(f) do
        begin
           read(f,x);
           while x>9 do
              x:=x div 10
           a[x]:=a[x]+1
        end;
      maxi:=a[0];
      for i:=0 to 9 do
          if a[i]>maxi then
            maxi:=a[i];
     for i:=0 to 9 do
          if a[i]=maxi then
            write(f,i,' ');
     write(f,maxi);
     close(f);
     close (g);

end.
```

Limbajul C/C++

```
#include<iostream>
#include<fstream>
using namespace std;

int a[11],x,maxi,i;

int main()
{   ifstream f("bac.in");
    ofstream g("bac.out");
    while (f>>x)
    {
        while (x>9)
            x=x/10;
        a[x]++;
    }
    maxi=a[0];
    for (i=1;i<=9;i++)
        if (a[i]>maxi)
        maxi=a[i];
    for (i=0;i<=9;i++)
        if (a[i]==maxi)
            g<<i<<" ";
    g<<maxi;
    return 0;
}
```

Se acordă 1p. pentru operaţiile cu fişiere (declarare, nume corect și deschidere pentru citire), încă 1p. pentru citirea tuturor numerelor din fişier, 1p. pentru un algoritm principial corect, încă 1p. pentru determinarea valorilor cerute, 1p. pentru afișarea lor, 1p. pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.).

Se acordă 1p. pentru alegerea unui algoritm eficient ca timp de executare (O(n)) și 1p. pentru utilizarea eficentă a memoriei. În total **8p.**

VARIANTA 24

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte

1. Variabilele **a** şi **b** sunt de tip întreg, **a=28**, iar **b** = **20**. Precizați care dintre expresiile de mai jos, are valoarea **TRUE,** respectiv valoarea **1**.

	Limbajul Pascal	Limbajul C/C++
a.	`3*a-4*b=0`	`3*a-4*b==0`
b.	`a mod 3 = b mod 3+2`	`a%3 == b%3+2`
c.	`NOT (b div 3+2=a div 3)`	`!(b/3+2==a/3)`
d.	`a-b-8<>0`	`a-b-8!=0`

2. Indicați răspunsul corect pentru apelul **s(7)**.

Limbajul Pascal

```
Var n:integer;
function
f(n:integer):longint;
begin
    if (n=1)
then  f:=1
else  f:=f(n-1)*n;
end;
function
s(n:integer):longint;
begin
    if (n=1)
 then  s:=f(1)
 else  s:=s(n-1)+f(n);
end;
```

Limbajul C/C++

```
long long f(int n)
{
    if (n==1)
       return 1;
    else
    return f(n-1)*n;
}
long long s(int n)
{
    if (n==1)
       return f(1);
    else
    return s(n-1)+f(n);
}
```

a. 5913	b. 873	c. 153	d. 33

3. Un arbore cu rădăcină are **7** noduri, numerotate de la **1** la **7**, și vectorul de tați **tata=(3,3,0,3,4,4,6)**. Indicați înălțimea arborelui, dacă se alege ca radăcină nodul **4**.

a. 2	b. 3	c. 4	d. 5

4.	Fie graful neorientat G=(V,M), unde V este mulțimea nodurilor grafului, card(V)=n, respectiv, M este mulțimea muchiilor grafului, iar card(M)=m. Graful dat are p componente conexe. Dacă n=12, m=21, iar p=10, precizați care este numărul maximal de cicluri independente care pot fi construite concomitent pe graf. Prin cicluri independente se înțelege, cicluri care conţin cel puţin câte o muchie care aparţine doar unuia din ele.			
	`a. 43`	`b. 23`	`c. 19`	`d. 10`
5.	Utilizând metoda backtracking se generează toate cuvintele formate din trei litere alăturate distincte, vocale litere mici. Cuvintele sunt generate în ordine lexicografică. Primele trei soluţii sunt: **aea**, **aei** şi **aeo**. Afişaţi ultimele trei soluţii generate.			
	`a. uoa,uoe,uoi`	`b. ueo,uio,uou`	`c. uoe,uoi,uou`	`d. uui,uuo,uuu`

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **`x%y`** restul împărţirii numărului întreg **`x`** la numărul întreg nenul **`y`** şi cu **`[a]`** partea întreagă a numărului real **`a`**.

```
citeste n,a (numar natural)
┌ pentru i←2,n execută
|   c←a
|   citeste b
|   d←b
| ┌ cât timp (c<>d)execută
| | dacă c>d atunci c←c-d
| | altfel d←d-c
| └■
| ┌ dacă d=1 atunci scrie b,' '
| └■
| a←b
└■
```

a)	Precizați valoarea afişată, dacă se citesc în ordine valorile **5 2023 51 34 85 289.** **(6p.)**
b)	Scrieţi un set de numere naturale, astfel încât, pentru **n=4,** în urma executării algoritmului, să se afişeze valoarea **23**. **(6p.)**
c)	Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **pentru...** cu structura **repetă ... până când**. **(6p.)**
d)	Scrieţi programul Pascal/C/C++ corespunzător algoritmului dat. **(10p.)**

2.	Variabila **s** memorează un șir de cel mult **31** de caractere. Scrieți secvența de instrucțiuni care numără câte grupuri de două vocale alăturate are textul. **Exemplu:** Pentru șirul s=`"D`**`au`**` examen de bacal`**`au`**`r`**`ea`**`t în ses`**`iu`**`n`**`ea`**` din `**`iu`**`l`**`ie`**`";` se va afișa `7`. **(6p.)**
3.	Fiecare dintre variabilele A și B, declarate alăturat, memorează coordonatele (x abscisa, iar y ordonata) câte unui punct în sistemul de coordonate xOy. Scrieți o expresie C/C++ care are valoarea 1 dacă și numai dacă segmentul cu extremitățile în punctele corespunzătoare variabilelor A și B intersectează axa Oy a sistemului de coordonate. **(6p.)** `struct punct { int x,y; }A,B;`

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Scrieţi definiţia completă a unui subprogram **zu**, cu doi parametri, care primeşte prin intermediul parametrilor: • **n**, un număr natural nenul (**0 < n < 100**); • **v**, un tablou unidimensional care memorează un şir de **n** numere naturale, fiecare având cel mult nouă cifre. Subprogramul **zu** determină modificarea şirului de numere primit prin parametrul **v,** realizând interschimbarea cifrei unităților cu cifra zecilor, pentru fiecare număr de cel puţin două cifre. Subprogramul va furniza prin parametrul **v,** tabloul modificat. Dacă în şirul primit nu există niciun număr de cel puţin două cifre, atunci parametrul **v** nu se va modifica. **Exemplu**. Pentru valorile **n=7**, **v=(31, 17, 2002, 1974, 7, 2021, 19)**, în urma apelului, subprogramului **zu** va furniza prin parametrii indicaţi valorile **n=7** şi **v=(13, 71, 2020, 1947, 7, 2012, 91).** **(10p.)**
2.	Să se construiască o matrice cu **n** linii şi **m** coloane care conţine primii **nxm** multiplii impari ai lui **5**. Scrieţi programul care citeşte de la tastatură două valori naturale **n** şi **m** (2<**n**<31, 2<**m**<17), construieşte matricea conform cerinţei şi o afişează pe ecran, pe linii, cu spaţii între elementele de pe fiecare linie. **Exemplu:** dacă se citeşte de la tastatură **n=3** şi **m=4**, se va construi şi se va afişa pe ecran matricea: 5 15 25 35 45 55 65 75 85 95 105 115 **(10p.)**
3.	Fişierul text **centenar.in** conţine un şir **a** de cel mult un milion de naturale nenule, formate fiecare din cel mult **9** cifre, separate prin câte un spaţiu. **a**) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină și afișează pe ecran numerele centenar din șir. Un număr se numește **centenar**, dacă prima cifră a numărului, înmulțită cu numărul format din ultimele două cifre ale aceluiaș număr, este **100**. Spunem că un număr **a**bc**de** este centenar dacă **a*de=100** **(8p.)** **Exemplu:** dacă fişierul **centenar.in** are conţinutul: **250 20025 4025 50120 7120 520 412325** atunci, pe ecran se va afişa **250 4025 50120520 412325** reprezentând numerele centenar din șir. **b)** Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

VARIANTA 24 - *Rezolvare*

Subiectul 1 - rezolvare

1.	`NOT(b div 3 + 2 = a div 3)`	`!(b/3+2==a/3)`
	Pentru precizarea răspunsului **c)** se acordă **4p**.	
2.	`5913` Pentru răspunsul **a)** se acordă **4p**.	
3.	Se observă că pentru rădăcina 4, vectorul de tați este, `tata=(3,3,4,0,4,4,6)`,iar înălțimea arborelui este 2. Pentru răspunsul **a)** se acordă **4p**.	
4.	Numărul ciclomatic m-n+p=19 Pentru răspunsul **c)** se acordă **4p.**	
5.	`uoe,uoi,uou` Pentru precizarea răspunsului **c)** se acordă **4p**.	

Subiectul 2 - rezolvare

1.	**a)**	Valoarea afișată, este 17,dacă se citesc în ordine valorile **5 2023 51 34 85 289.** Algoritmul determină cel mai mare divizor comun dintre numerele date. Pentru precizarea valorii `17` se acordă **6p**.
	b)	Orice 4 valori prime între ele. De exemplu: (1,2,3,4) Pentru precizarea unui set de valori corecte se acordă **6p**
	c)	Pentru scrierea corectă a unei structuri repetitive se acordă 2p., pentru testarea inițială a valorii lui **`n`** se acordă 2p., pentru testarea condiției de continuare sau oprire se acordă 1p, iar pentru scrierea integrală a restului algoritmului se mai acordă 1p. - în total **6p**.
	d)	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc. Pentru instrucțiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului și de decizie se acordă câte 1p., pentru instrucţiunea repetitivă se acordă 3p., iar pentru cele 6 atribuiri se acordă 2p. Pentru structura corectă a programului se mai acordă 1p., în total **10p**.
2.	**Un răspuns corect:** se numără câte poziții **i**, respectiv **i+1** au caractere vocale. Numărul astfel caculat se afișează. Pentru orice răspuns corect se acordă **6p**.	
3.	Un răspuns corect: A.x*B.x<=0 Pentru orice răspuns corect se acordă **6p**.	

Subiectul 3 - rezolvare

<table>
<tr><td>1.</td><td>

Prelucrarea oferită de subprogram verifică existența celor două cifre în scrierea fiecărui număr memorat în tablou. Ștergerea unui element care conține cifra **a** și nu conține cifra **b** se realizează prin deplasarea cu o poziție la stânga a tuturor elementelor situate la dreapta celui ce urmează a fi șters. După realizarea deplasării, se micșorează valoarea lui **n** cu **1**.

Limbajul Pascal

```
type vector=array[1..100]of longint;
Var n,i:byte;
    v:vector;
procedure sub(n:byte; var v:vector);
var u,z,i:byte;
begin

        for i:=1 to n do
        if v[i]>9 then
                begin
                        u:=v[i] mod 10;
                        z:=(v[i] div 10) mod 10;
                        v[i]:=((v[i] div 100)*10+u)*10+z;
        end;
        writeln();
end;
```

Limbajul C/C++

```
void zu(int n, long v[101])
{
    for (int i=1;i<=n;i++)
       if(v[i]>=10)
       {int u=v[i]%10;
        int z=v[i]/10%10;
        v[i]=((v[i]/100)*10+u)*10+z;
       }
}
```

Pentru antet corect se acordă 3p. (Pentru respectarea structurii antetului (procedure/void) se acordă 2p., pentru declararea corectă a parametrilor **n** și **v** încă 2p.).

Pentru declararea variabilelor locale încă 1p., iar pentru respectarea structurii subprogramului și a sintaxei limbajului se acordă încă 1p.

Pentru determinarea elementelor cifrei unităților și a zecilor câte 1p. Pentru modificarea tabloului cu toate elemente ce memorează numerele cu proprietatea cerută 2p. Pentru corectitudinea globală a programului 1p.

În total **10p.**

</td></tr>
<tr><td>2</td><td>

Pot fi concepute mai multe metode de obținere a matricei cerute. Metoda clasică ce completează coloanele matricei (numerotate de la **1** la **m**), respectiv liniile (numerotate de la **1** la **n**), în maniera descrisă, este:

Limbajul Pascal

```
Var n,m,x,i,j:integer;
a:array[1..101,1..101] of integer;
```

</td></tr>
</table>

```
begin
  x := 5;
  readln(n,m);
  for i := 1 to n do
    for j := 1 to m do
    begin
    a[i,j] := x;
      x := x+10;
    end;
  for i := 1 to n do
  begin
    for j := 1 to m do
      write(a[i,j],' ');
      writeln;
    end
end.
```

Limbajul C/C++

```
#include <iostream>
using namespace std;
int main ()
{
  int n, m, x = 5;
  cin >> n >> m;
  int a[n + 1][m + 1];
  for (int i = 1; i <= n; i++)
    for (int j = 1; j <= m; j++)
      {
      a[i][j] = x;
      x += 10;
      }
  for (int i = 1; i <= n; i++)
    {
      for (int j = 1; j <= m; j++)
      cout << a[i][j] << " ";
      cout << endl;
    }
  return 0;
}
```

Pentru declararea corectă a matricei se acordă 1p., pentru citirea lui **n** și **m** se mai acordă 1p., pentru parcurgerea unei coloane sau linii se acordă 1p., pentru completarea corectă a unei coloane se acordă 2p., pentru completarea corectă a tuturor coloanelor se acordă 2p., iar pentru afișarea matricei conform cerinței se acordă 2p. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă 1p. - în total **10p**.

3. **b)**

Algoritmul de rezolvare propus citește, unul câte unul, numerele din fișier, analizând proprietatea de centenar a fiecărui număr și afișarea acestuia, în caz afirmativ.

Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul. Deoarece numărul de valori din fișier (1000000) este dimensiunea care contează în problemă, spunem că am obținut un algoritm liniar. Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple.

		Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. - în total **2p.**
	a)	Realizez un subprogram care testează dacă un număr arbitrar este centenar.

Limbajul Pascal

```
var f:text;
x:longint;

function centenar(x:longint):byte;
var zu:integer;
begin
 zu:=x mod 100;
 while x>9 do
   x:= x div 10;
 if x*zu = 100 then centenar:= 1
 else centenar := 0;
end;

begin
 assign(f,'centenar.in');reset(f);
 while not eof(f) do
  begin
   read(f,x);
   if centenar(x)=1 then write(x,' ');
  end;
 close(f);
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;

int centenar(int x)
{ int zu=x%100;
  while(x>9)
  x=x/10;
  return (x*zu==100);
}

int main()
{ int x;
  ifstream fin("centenar.in");
  while(fin>>x)
    if(centenar(x)==1)
       cout<<x<<" ";
  return 0;
}
```

VARIANTA 25

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Stabiliți care poate fi cea mai mare valoare a expresiei **Pascal/C/C++** alăturată știind că variabila **x** memorează un număr întreg.

`22 + x%2000`

a. 22	b. 23	c. 2022	d. 2021

2. Pentru subprogramul **f** cu definiția următoare, indicați valoarea care se va afișa la apelul **f(20)**.

Limbajul Pascal

```
function f(x:integer): integer;
begin
  if x=0 then f:=0
  else
    if x mod 2 = 0 then f:=f(x-1)
    else f:=x*x+f(x div 2);
end;
```

Limbajul C/C++

```
int f(int x)
{
 if(x==0) return 0;
 if(x%2==0) return f(x-1);
 return x*x+f(x-1);
}
```

a. 55	b. 100	c. 165	d. 210

3. Utilizând metoda backtracking se generează în ordine crescătoare toate numerele naturale de câte patru cifre distincte din mulțimea **{1,2,3,4,5}**. Numerele generate nu conțin două cifre pare alăturate. Primele șase numere generate sunt, în această ordine: **1234**, **1235**, **1243**, **1245**, **1253**, **1254**. Stabiliți câte dintre numerele generate au prima și ultima cifră pare.

a. 3	b. 6	c. 12	d. 18

4. Se consideră arborele cu **9** noduri, etichetate cu numere distincte de la **1** a **9**, dat prin următorul vector de taţi: **(5,7,7,5,0,7,5,6,6)**. Stabiliți care este lungimea celui mai lung lanț din arborele dat:

a.4	b. 6	c. 8	d. 9

5. Se consideră un graf neorientat cu **1000** de componente conexe. Indicați care este numărul minim de muchii care pot fi adăugate grafului dat astfel încât să devină conex.

a. 199	b. 200	c. 999	d. 1000

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. **Se consideră algoritmul următor reprezentat în pseudocod:**

S-a notat cu **x%y** restul împărţirii numărului întreg **x** la numărul întreg nenul **y** şi cu **[a]** partea întreagă a numărului real **a**.

```
citeşte a (număr natural)
d←0; b←0; p←1
┌cât timp a>0 execută
│ c←a%10
│ ┌pentru i←1,c execută
│ │    d←(d+1)%10
│ └■
│ b←b+d*p; a←[a/10]; p←p*10
└■
scrie b
```

a) Scrieţi ce valoare va fi afişată dacă se citeşte valoarea 12345? **(6p.)**

b) Scrieţi un număr natural, format din cinci cifre, care poate fi citit pentru variabila **a** astfel încât, să se afişeze valoarea 56789. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **pentru...execută** cu o instrucţiune de atribuire. **(6p.)**

d) Scrieţi programul **Pascal/C/C++** corespunzător algoritmului dat. **(10p.)**

2. Se consideră declarările următoare în care variabilele **S1** şi **S2** memorează informaţiile referitoare la un câte un segment din planul cu sistemul de coordonate **xOy**: coordonatele capetelor segmentului în sistemul de coordonate **xOy**. Scrieţi o expresie în limbajul **Pascal|C/C++** care să aibă valoarea **true/1** dacă şi numai dacă cele două segmente coincid. **(6p.)**

Limbajul Pascal

```
Type punct=record
           x,y:integer
end;
segment = record
          A,B:punct;
end;
var S1,S2: segment;
```

Limbajul C/C++

```
struct punct {
        int x,y;
};
struct segment {
        struct punct A,B;
};
segment S1, S2;
```

3. În secvenţa de program alăturată, variabila **s** memorează un şir cu cel mult 30 de caractere (litere, cifre şi spaţii), iar variabilele **i** şi **sum** sunt de tip întreg. Completaţi punctele de suspensie din secvenţă astfel încât, în urma executării secvenţei, aceasta să afişeze un număr natural reprezentând suma tuturor cifrelor din şirul **s**. **(6p.)**

Limbajul Pascal

```
sum:=0;
for i:=1 to length(s) do
  if ...... then sum:=......;
write(sum);
```

Limbajul C/C++

```
sum=0;
for(i=0;i<strlen(s);i++)
    if(......)sum=......;
cout<<sum;
```

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Scrieţi în caseta text următoare definiţia completă a unui subprogram **sub**, cu doi parametri, care primeşte prin intermediul parametrilor:

- **n**, un număr natural nenul (**0 $<$ n $<$ 100**);
- **v**, un tablou unidimensional ce memorează un şir de **n** numere naturale strict mai mari decât **1**, fiecare având cel mult patru cifre.

Subprogramul **sub** înlocuieşte fiecare număr, din şirul de numere primit prin parametrul **v**, cu cel mai mic divizor al său diferit de **1.** De asemenea subprogramul va furniza prin intermediul parametrului **v** tabloul modificat. De exemplu, pentru valorile `n=5`, `v=(35,7,52,121,13)`, ale parametrilor, în urma apelului, subprogramului **sub** va furniza prin parametrul **v** tabloul: `(5,7,2,11,13)`. **(10p.)**

2. Scrieți un program **Pascal|C/C++** care să citească de la tastură numărele naturale `n` și `m` (`3<n,m<25`), apoi să construiască o matrice cu `n` linii și `m` coloane care conţine doar valori din mulţimea `{1,2,3}` și cu proprietatea că nu există două valori egale alăturate pe linie sau pe coloană. Matricea construită se va afișa pe ecran, pe linii, cu spații între elementele de pe fiecare linie. De exemplu, pentru `n=3` și `m=5`, o soluție poate fi matricea alăturată: **(10p.)**

```
1 2 3 2 3
2 3 1 3 1
3 1 2 1 2
```

3. Fişierul text bac.in conţine un şir **s** de cel mult un milion de numere naturale nenule, formate fiecare din cel mult **5** cifre, separate prin câte un spaţiu.

a) Scrieţi un program care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină și afișează pe ecran lungimea maximă a unei secvenţe de numere din şir cu proprietatea că valorile învecinate au parităţi diferite. O secvenţă a unui şir constă în elemente situate pe poziţii consecutive în şirul considerat. **(8p.)**

Exemplu: dacă fişierul **bac.in** are conţinutul:

```
10 12 11 10 2 33 4 13 39 51 3 4 31 8 6 81
```

atunci, pe ecran se va afişa numărul **4 (**lungimea maximă a unei secvenţe formată doar din numere învecinate cu parități diferite.

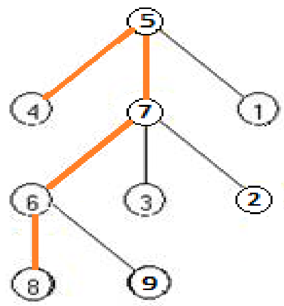
b) Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

Observaţii:

- Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
- Timpul efectiv de lucru este de 3 ore.
- În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
- În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 25 - *Rezolvare*

Subiectul 1

<table>
<tr><td>1.</td><td colspan="2">Pentru precizarea răspunsului <code>d)</code> se acordă 4p.</td></tr>
<tr><td>2.</td><td colspan="2"><code>f(10)=f(9)=81+f(8)=81+f(7)=81+49+f(6)=130+f(5)=</code>
<code>130+25+f(4)=155+f(3)= 155+9+f(2)=164+f(1)=165</code> $(=1^2+3^2+5^2+7^2+9^2)$
Pentru răspunsul c) se acordă 4p.</td></tr>
<tr><td>3.</td><td colspan="2">Numerele cerute sunt de forma: <code>2xy4</code> sau <code>4xy2</code> cu <code>xy</code> din mulțimea <code>{13,31,15,51,35,53}</code>. Astfel, sunt generate <code>12</code> numere cu proprietatea cerută.
Pentru răspunsul c) se acordă 4p.</td></tr>
<tr><td>4.</td><td>Din vectorul de referințe ascendente deducem că rădăcina arborelui este nodul 5 (tata[5]=0). În imaginea alăturată este reprezentat arborele dat prin vectorul de tați din enunt.
Observăm că lungimea celor mai lungi lanțuri elementare din arbore este <code>4</code>.
Pentru precizarea răspunsului a) se acordă 4p.</td><td></td></tr>
<tr><td>5.</td><td colspan="2">Sunt necesare doar <code>1000-1</code> muchii noi care să unească prima componentă conexă cu a doua, a doua cu a treia,..., ultima cu penultima. Pentru precizarea răspunsului <code>c)</code> se acordă 4p.</td></tr>
</table>

Subiectul 2

<table>
<tr><td>1.</td><td>a)</td><td colspan="2">Se construiește numărul b astfel încât cifra unităților este a%10. cifra zecilor este egală cu ultima cifră a sumei ultimelor două cifre ale lui a, cifra sutelor este egală cu cu ultima cifră a sumei ultimelor trei cifre ale lui a, ..., prima cifră cu ultima fiecare a sumei tuturor cifrelor lui a. Pentru precizarea valorii <code>54295</code> se acordă 6p.</td></tr>
<tr><td></td><td>b)</td><td colspan="2">Răspuns corect: <code>99999</code>. Pentru precizarea celor trei valori corecte se acordă 6p.</td></tr>
<tr><td></td><td>c)</td><td>Înlocuim structura pentru...execută cu atribuirea d←(d+c)%10. Pentru înlocuirea structuri repetitive cu o instrucțiune de atribuire se acordă 2p., pentru echivalența algoritmilor se acorda acordă 2p., iar pentru corectitudinea globală a algoritmului înlocuit se mai acordă 2p. - în total 6p.</td><td><code>citește a (număr natural)</code>
<code>d←0; b←0; p←1</code>
<code>┌cât timp a>0 execută</code>
<code>| c←a%10; d←(d+c)%10</code>
<code>| b←b+d*p; a←[a/10]; p←p*10</code>
<code>└■</code>
<code>scrie b</code></td></tr>
</table>

	d)	Pentru instrucţiunile corecte de declarare a variabilelor, de citire a datelor, de afişare a rezultatului şi de decizie se acordă câte **1p**., pentru cele două instrucţiuni repetitive se acordă **3p**., iar pentru cele **6** atribuiri se acordă **2p**. Pentru structura corectă a programului se mai acordă **1p.**, în total **10p**.

2. Pentru orice răspuns corect se acordă **6p**. Un răspuns corect:

C/C++

```
S1.A.x==S2.A.x && S1.A.y==S2.A.y && S1.B.x==S2.B.x  && S1.B.y==S2.B.y ||
S1.A.x==S2.B.x && S1.A.y==S2.B.y && S1.B.x==S2.A.x && S1.B.y==S2.A.y
```

Pascal

```
S1.A.x=S2.A.x and S1.A.y=S2.A.y and S1.B.x=S2.B.x and S1.B.y=S2.B.y or
S1.A.x=S2.B.x and S1.A.y=S2.B.y and S1.B.x=S2.A.x and S1.B.y=S2.A.y
```

3. Pentru orice răspuns se acordă 6**p** (câte 3p. pentru fiecare înlocuite corectă a punctelor de suspensie). Un răspuns corect:

C/C++

```
sum=0;
for(i=0;i<strlen(s);i++)
  if(isdigit(s[i]))sum=sum+(s[i]-'0');
cout<<sum;
```

Pascal

```
for i:=1 to length(s) do
  if s[i] in ['0'..'9']  then sum:=sum+ord(s[i])-ord('0');
writeln(sum);
```

Subiectul 3

1. Prelucrarea oferită de subprogram determină poziția primei apariții a valorii minime din tablou, poziția ultimei apariții a valorile maxime în șir, apoi interschimbă cele două valori situate pe pozițiile determinate. Tabloul rezultat este returnat prin intermediul parametrului **v** al subprogamului.

Limbajul Pascal	Limbajul C/C++
`Type vector=array[1..100]of integer;` `procedure sub(n:byte;var v:vector);` `var i:byte;` `    d:integer;` `begin for i:=1 to n do begin` `        d:=2;` `        while v[i]%d <> 0 then inc(d);` `        v[i]:=d` `        end;` `end;`	`void sub(int n, int v[])` `{` `    int i, d;` `    for(i=1; i<=n; i++)` `       { d=2;` `         while(v[i]%d!=0)d++;` `         v[i]=d;` `       }` `}`

Pentru antet corect se acordă **2p**. Pentru determinarea unui divizor al unui element din tabloul **v** se acordă **1p.**, iar pentru determinarea celui mai mic divizor pt un element din tablou se acordă **1p**. Pentru inlocuirea corectă a unui element din tablou conform cerinței se acordă **2p.**, iar pentru înlocuirea tuturor elementelor tabloului conform cerinței se acordă **3p**. Pentru corectitudinea globală a programului **1p**.

2.	Pot fi concepute mai multe metode de obținere a matricei cerute. Metoda clasică ce completează coloanele matricei (numerotate de la **1** la **m**), respectiv liniile (numerotate de la **1** la **n**), în maniera descrisă, este cea din imaginea alăturată. Pentru declararea corectă a matricei se acordă **1p**., pentru citirea valorilor **n** și **m** se mai acordă	<pre>k←1 ┌pentru i←1,n execută │ ┌pentru j←1,m execută │ │ A[i][j]←k; k←k+1 │ │ ┌dacă k>3 atunci k←1 │ │ └■ │ └■ └■</pre>
	1p., pentru parcurgerea unei linii sau coloane se acordă **1p**., pentru completarea corectă a unei linii, respectiv coloane, se acordă câte **1p**., pentru completarea corectă a tuturor valorilor se acordă **2p**., iar pentru afișarea matricei conform cerinței se acordă **2p**. În plus, pentru declararea variabilelor simple, structura și corectitudinea sintactică a programului se mai acordă **1p**. - în total **10p**.	

3. b) Algoritmul de rezolvare propus determină paritatea primului număr și apoi citește un număr din fișier, comparând paritatea acestuia cu cea a anteriorului citit: dacă sunt diferite, se "prelungește" secvența curentă. În caz contrar, se trece la o nouă secvență. La finalul oricărei secvențe se analizează lungimea ei, în vederea memorării lungimii maxime. Eficiența algoritmului, ca timp de executare, constă în faptul că determinăm valoarea cerută în timpul citirii datelor, deci parcurgem o singură dată șirul de numere (O(n)). Ca spațiu de memorie, soluția propusă este eficientă, deoarece utilizează doar variabile simple, fără folosirea unor structuri suplimentare de date (O(1)).

Pentru o descriere coerentă a metodei se acordă **1p**., iar pentru justificarea eficienței, încă **1p**. - în total **2p**.

a)

Limbajul Pascal

```
var f:text;
    lgmax,lg,x,y:longint;
begin
 assign(f,'bac.in');reset(f);
 read(f,x);
 lg:=1;
 while not(eof(f)) do begin
    read(f,y);
    if x mod 2 <> y mod 2
      then inc(lg)
      else begin
             if lgmax<lg
                then lgmax:=lg;
             lg:=1 end;
    x:=y;
 end;
 if lgmax<lg then lgmax:=lg;
 close(f);  write(lgmax)
end.
```

Limbajul C/C++

```
#include <iostream>
#include <fstream>
using namespace std;

ifstream f("bac.in");
int main()
{ int lgmax=0, lg=1, x, y;
  f>>x;
  while(f>>nr)
  {
   if (x%2 != y%2) lg++;
     else
     { if(lgmax<lg) lgmax=lg;
       lg=1;
     }
  }
  if (lgmax<lg) lgmax=lg;
  cout<<lgmax;
  return 0;
}
```

Se acordă **1p**. pentru operaţiile cu fişiere (declarare, nume corect şi deschidere pentru citire), încă **1p**. pentru citirea tuturor numerelor din fişier, **1p.** pentru un algoritm principial corect, încă **1p.** pentru determinarea valorii cerute, **1p.** pentru afișarea ei, **1p.** pentru corectitudinea formală (declararea variabilelor, structura programului, sintaxa instrucțiunilor, etc.). Se acordă **1p.** pentru alegerea unui algoritm eficient ca timp de executare (**O(n)**) și **1p.** pentru utilizarea eficentă a memoriei. În total, **8p**.

VARIANTA 26

Subiectul 1 (20p)

Pentru fiecare dintre itemii de la 1 la 5, scrieţi pe foaia de examen litera corespunzătoare răspunsului corect. Fiecare răspuns corect se notează cu 4 puncte.

1. Se consideră un număr natural **x** cu exact **5** cifre. Indicați care dintre următoarele expresii elimină din numărul **x** cifra din mijloc:

	Limbajul Pascal	Limbajul C/C++
a.	`x div 1000*100+x mod 100`	`x/1000*100+x%100`
b.	`x div 1000+x mod 100`	`x/1000+x%100`
c.	`x div 1000+x mod 100*100`	`x/1000+x%100*100`
d.	`x div 1000+ x mod 1000`	`x/1000+x%1000`

2. Pentru subprogramul **fct** cu definiţia următoare, indicaţi valoarea **fct(1203456)**.

Limbajul Pascal

```
function
fct(a:longint):longint;
begin
if  a<=9 then
  if a mod 2 = 0 then fct:= a
              else fct:=0
  else
 if a mod 2 = 0 then
fct:=10*fct(a div 10)+a mod 10
        else fct:=fct(a div 10)
end;
```

Limbajul C/C++

```
int fct(int a)
{
 if (a<=9)
 if (a % 2 == 0) return a;
              else return 0;
 if (a% 2 == 0)
return 10*fct(a/10)+a%10;
return fct(a/10);
}
```

a. 135	b. 2046	c. 246	d. 6402523

3. Utilizând metoda backtracking, se generează toate submulţimile cu **6** elemente (elementele fiecărei submulţimi fiind ordonate crescător) ale mulţimii **{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20}**. Indicaţi numărul soluțiilor care conțin elementele **4, 5, 6**.

a. 210	b. 680	c. 20	d. 40

4. Indicați care este matricea de adiacență a unui graf neorientat cu 5 noduri, 3 muchii și cel puțin un nod izolat.

a.	b.	c.	d.
0 1 0 1 0 1 0 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0	1 1 0 1 0 1 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0	0 1 0 1 0 1 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0	0 1 0 1 0 1 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0

5.	Se consideră un arbore cu **7** noduri. Indicaţi valoarea variabilei **x** astfel încât şirul următor **2,1,3,2,1,1,x** să reprezinte gradele nodurilor arborelui.		
	a. 2	b. 0	c. 3	d. 1

Subiectul 2 (40p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.

1. Se consideră algoritmul următor reprezentat în pseudocod:

```
citeşte x,y,z (numere naturale nenule)
┌cât timp x≠y || x≠z || y≠z execută
| t<-x
| ┌dacă y<t atunci t<-y
| └■
| ┌dacă z<t atunci t<-z
| └■
| ┌dacă x≠t atunci x<-x-t
| └■
| ┌dacă y≠t atunci y<-y-t
| └■
| ┌dacă z≠t atunci z<-z-t
| └■
└■
scrie x
```

a) Ce valoare va fi afişată dacă se citesc numerele **120,56,48**? **(6p.)**

b) Scrieţi trei numere distincte din intervalul **[100,1000)** astfel încât să se afişeze valoarea **6**. **(6p.)**

c) Scrieţi în pseudocod un algoritm, echivalent cu cel dat, în care să se înlocuiască structura **cât timp...execută** cu o structură repetitivă de alt tip. **(6p.)**

d) Scrieţi programul **Pascal/C/C++** corespunzător algoritmului dat. **(10p.)**

2. Se consideră structura **triunghi**, care memorează numere reale reprezentând lungimile laturilor (**a,b,c**) , semiperimetrul (**p**) şi aria (**s**) unui triunghi, iar variabila **t** este de tipul **triunghi**:

Limbajul Pascal	Limbajul C/C++
`type triunghi = record a,b,c,p,s:real end;` `var t: triunghi;`	`struct triunghi{ float a,b,c,p,s; }t;`

Scrieţi secvenţa de instrucţiuni **Pascal/C/C++** care calculează şi afişează aria triunghiului. **(6p.)**

3.	În secvenţa alăturată, variabila **a** memorează elementele unui tablou bidimensional cu **6** linii (numerotate de la 1 la 6) şi **6** coloane (numerotate de la 1 la 6), iar toate celelalte variabile sunt de tip întreg. Indicaţi valoarea variabilei **p** în urma executării secvenţei de program. **(6p.)**	
	Limbajul Pascal	**Limbajul C/C++**
	`for i:=1 to 6 do` `    for j:=1 to 6 do` `       a[i,j]:=(i-1)*6+j;` `p:=0;` `for i:=1 to 6 do` ` if a[i,7-i] mod 2 <>0 then` `       p:=p+a[i,i];`	`for(i=1;i<=6;i++)` `    for(j=1;j<=6;j++)` `       a[i][j]=(i-1)*6+j;` `p=0;` `for(i=1;i<=6;i++)` `   if(a[i][7-i]%2 != 0)` `        p=p+a[i][i];`

Subiectul 3 (30p)

Scrieţi pe foaia de examen răspunsul pentru fiecare dintre cerinţele următoare.	
1.	Subprogramul **verifica** primeşte prin intermediul parametrilor **a** şi **b** două numere naturale din intervalul [100,10^6] şi returnează valoarea **1** dacă numărul **b** se poate obţine din numărul **a** prin eliminarea unei singure cifre (fără a schimba ordinea cifrelor) şi valoarea **0**, în caz contrar. Scrieţi definiţia completă a subprogramului **verifica**. **Exemplu:** pentru **a=1234** şi **b=134**, subprogramul va returna valoarea **1**. pentru **a=1234** şi **b=143**, subprogramul va returna valoarea **0**. **(10p.)**
2.	Într-un text cu cel mult **100** caractere, cuvintele sunt formate din litere mici ale alfabetului englez şi sunt separate prin câte un spaţiu. Scrieţi un program **Pacal /C/C++** care citeşte de la tastatură un text de tipul menţionat şi construieste în memorie un şir de caractere care conţine doar cuvintele alcătuite din caractere distincte, cuvintele şirului fiind separate prin câte un spaţiu. Şirul obţinut este afişat pe ecran, iar dacă nu există astfel de cuvinte, se afişează pe ecran mesajul **nu exista**. **(10p.)** **Exemplu**: dacă textul citit este **caietul este de matematica** se afişează **caietul de** dacă textul citit este **este aici** se afişează **nu exista**
3.	Cifra de control a unui număr natural se obţine efectuând suma cifrelor sale, apoi suma cifrelor acestei sume, până când suma obţinută este un număr format dintr-o singură cifră. Această ultimă cifră poartă numele de cifră de control. Cifra de control (cifc) a numărului n=123456: cifc(123456)=cifc(21)=cifc(3)=3. Fişierul text **bac.in** conţine un şir cu cel mult **1000000** de numere naturale cu valori de cel mult 8 cifre, valori separate prin spaţii. a) Scrieţi un program Pascal/C/C++ care, utilizând un algoritm eficient din punct de vedere al timpului de executare şi al spaţiului de memorie utilizat, determină şi afişează pe ecran elementul maxim din fişier şi câte numere din fişier au cifra de control egală cu a elementului maxim. **(8p.)**

	Exemplu: dacă fişierul **date.in** are conţinutul: **123 60 51 321 111 120 501 89 301 12 13 134** atunci se afişează **501 5** b) Descrieţi în limbaj natural metoda utilizată şi explicaţi în ce constă eficienţa ei. **(2p.)**

Observaţii:

• Toate subiectele sunt obligatorii. Se acordă 10 puncte din oficiu.
• Timpul efectiv de lucru este de 3 ore.
• În rezolvările cerute, identificatorii utilizaţi trebuie să respecte precizările din enunţ (bold), iar în lipsa unor precizări explicite, notaţiile trebuie să corespundă cu semnificaţiile asociate acestora (eventual în formă prescurtată).
• În programele cerute, datele de intrare se consideră corecte, validarea acestora nefiind necesară.

VARIANTA 26 - *Rezolvare*

Subiectul 1

1.	Pentru precizarea răspunsului **a)** se acordă **4p**.
2.	Pentru răspunsul **b)** se acordă **4p**. Indicații: functia returnează numărul format cu cifrele pare ale lui a.
3.	Pentru răspunsul **b)** se acordă **4p**. Numărul submulţimilor de forma **4,5,6,x,y,z** este C_{14}^{3} Numărul submulţimilor de forma **x,4,5,6,y,z** este 3*C_{14}^{2} Numărul submulţimilor de forma **x,y,4,5,6,z** este 3*14 Numărul submulţimilor de forma **x,y,z,4,5,6** este 1
4.	Pentru răspunsul **d)** se acordă **4p**.
5.	Pentru precizarea răspunsului **a)** se acordă **4p**. Un arbore cu **7** noduri are **6** muchii, iar suma gradelor nodurilor arborelui este egală cu **12**.

Subiectul 2

1.	**a)**	Răspuns corect `8`. **6p**.
	b)	Algoritmul determină cel mai mare divizor comun al numerelor **x,y,z**. Un set de date de intrare pentru care se afişează valoarea `6` este: `150, 294, 132`. Se acordă câte **2p.** pentru fiecare număr care respectă cerinţa. **6p**.
	c)	(vezi pseudocodul de mai jos)

```
  citeşte x,y,z (numere naturale nenule)
┌ dacă x≠y || x≠z || y≠z atunci
│┌repetă
││  t<-x
││ ┌dacă y<t atunci t<- y
││ └■
││ ┌dacă z<t atunci t<- z
││ └■
││ ┌dacă x≠t atunci x<- x-t
││ └■
││ ┌dacă y≠t atunci y<- y-t
││ └■
││ ┌dacă z≠t atunci z<- z-t
││ └■
│└până când x=y && x=z && y=z
└■
  scrie x
```

		Pentru echivalenţa prelucrării realizate conform cerinţei se acordă **5p**, iar pentru corectitudinea globală a algoritmului se acordă **1p**. Se acordă numai **2p**. dacă algoritmul are o structură repetitivă conform cerinţei, principial corectă, dar nu este echivalentă cu cea dată.
	d)	Se testează capacitatea de a reprezenta pe hârtie algoritmul dat cu ajutorul unui limbaj de programare studiat. Deși scrierea programelor pe hârtie este o activitate improprie la informatică, prin această cerință se valorifică experiențele anterioare de implementare și testare a programelor pe calculator, privind structura programelor, declararea variabilelor, sintaxa instrucțiunilor programului, regulile de scriere a expresiilor, etc. Pentru declararea variabilelor se acordă **1p.**, pentru citirea datelor şi afişarea rezultatului se acordă **1p**, pentru instrucţiunile de decizie se acordă **5p.**, pentru instrucţiunea repetitivă se acordă **2p.** şi pentru corectitudinea globală a pogramului se acordă **1p.**.
2.	**Limbajul Pascal** `t.p:=(t.a+t.b+t.c)/2;` `t.s:=sqrt(t.p*(t.p-t.a)*(t.p-t.b)*(t.p-t.c));` **Limbajul C/C++** `t.p=(t.a+t.b+t.c)/2;` `t.s=sqrt(t.p*(t.p-t.a)*(t.p-t.b)*(t.p-t.c));` Pentru accesul corect la câmpurile înregistrării se acordă **2p.**, iar pentru calcularea ariei se acordă **4p.**	
3.	Răspunsul corect: **66** şi se acordă **6p**.	

Subiectul 3

1. **Limbajul Pascal**

```
function verifica(a,b:integer):integer;
var ok,ca,x,p:integer;

begin
ok:=0;
p:=1;
ca:=a;
repeat
      x:=a div(p*10)*p+a mod p;
      p:=p*10;
      if x=b  then ok:=1;
      ca:=ca div 10;
until ca=0;
verifica:=ok;
end;
```

	Limbajul C/C++ `int verifica (int a, int b)` `{int ok,ca,x,p;` ` ok=0;` ` p=1;ca=a;` ` do{ x=a/(p*10)*p+a%p;` `     p=p*10;` `     if(x==b) ok=1;` `     ca=ca/10;` `    }while(ca );` `return ok;` `}` Pentru antetul subprogramului se acordă **3p.**, pentru determinarea valorii cerute se acordă **6p.**, pentru corectitudinea globală a subprogramului se acordă **1p.**
2.	**Limbajul Pascal**

```
var s,x,p:string;
    nr,i,ok,c,j:integer;
    v:array[0..25] of integer;

begin
 readln(s);
 s:=s+' ';
 nr:=0;i:=0;
 while length(s)>0 do
   if s[i]=' ' then
    begin
    ok:=1;
    for j:=0 to 25 do
            v[j]:=0;
     for j:=1 to i-1 do
        begin
            c:=ord(s[j])-ord('a');
            v[c]:=v[c]+1;;
            if v[c]>1 then ok:=0;
        end;
      if ok=1 then begin
                     nr:=nr+1;
                     if nr=1 then x:=copy(s,1,i-1)
                           else  x:=x+' '+copy(s,1,i-1);
                     end;
     delete(s,1,i);
     i:=0;
    end
    else i:=i+1;
    if nr=0 then write('nu exista')
             else write(x);
end.
```

Limbajul C/C++

```
#include <iostream>
# include<cstring>
using namespace std;
int main()
{   char s[101],x[101],*p;
    int nr=0,i,v[26],ok,c;
    cin.get(s,101);
    x[0]='\0';
    p=strtok(s," ");
    while(p)
    {
        ok=1;
        for(i=0;i<=25;i++)
            v[i]=0;
        for(i=0;i<strlen(p)&& ok;i++)
        {
            c=p[i]-'a';
            v[c]++;
            if(v[c]>1) ok=0;

        }
        if(ok){nr++;
               if(nr==1) strcat(x,p);
                   else {strcat(x," ");
                         strcat(x,p);
                          }
               }
     p=strtok(NULL," ");
    }
     if(nr==0) cout<<"nu exista";
              else cout<<x;
    return 0;
 }
```

Pentru citirea corectă a şirului de caractere se acordă **1p**., pentru determinarea unui cuvânt din componenţa şirului se acordă **3p**., pentru verificarea proprietăţii că are literele distincte se acordă **3p.**, pentru determinarea cazului în care nu există cuvinte cu proprietatea cerută se acordă **1p**., pentru construirea şirului se acordă **2p**., pentru corectitudinea globală a programului se acordă **1p.**

3. **a)**

Limbajul Pascal

```
var  v: array[0..9] of integer;
     maxx,x,c,cmax,i: longint;
     fin:text;
begin
 maxx:=-1;
```

```
 for i:=0 to 9 do v[i]:=0;
 assign(fin, 'bac.in');
 reset(fin );
 while not eof(fin)  do begin
   read(fin,x);
     if x>maxx then maxx:=x;
     if x=0 then c:=0
          else if x mod 9=0 then  c:=9
                      else c:=x mod 9;
    v[c]:=v[c]+1;
    if x=maxx then cmax:=c;
 end;
 write(maxx, ' ',v[cmax]);
 close(fin);
end.
```

Limbajul C/C++

```
#include <iostream>
# include<fstream>
using namespace std;
int v[10];
int main()
{ ifstream fin("date.in");
 int maxx,x,c,cmax;
 maxx=-1;
 while (fin>>x)
 {
     if(x>maxx)maxx=x;
     if(x==0)c=0;
          else if(x%9==0) c=9;
                      else c=x%9;
    v[c]++;
    if(x==maxx)cmax=c;
 }
 cout<<maxx<<" "<<v[cmax];
 fin.close();
    return 0;
}
```

Pentru operaţii cu fişiere, declarare, pregătire în vederea citirii, citire din fişier se acordă **1p**., pentru determinarea valorilor cerute se acordă **5p**., pentru utilizarea unui algoritm eficient se acordă **1p.**, pentru declararea variabilelor, afişarea datelor şi corectitudinea globală a programului se acordă **1p.**.

	b)	O soluție posibilă: Pe măsură ce citim numerele, determinăm elementul maxim (maxx), cifra de control a numărului curent (c), reţinem în variabila cmax cifra de control a numărului maxim şi contorizăm aparițiile cifrei de control (v[c]). Eficiența algoritmului, ca timp de executare, constă în faptul că parcurgem o singură dată șirul de numere. Deoarece numărul de valori din fișier (1000000) este dimensiunea care contează în problemă, spunem că am obținut un algoritm liniar. Ca spațiu de memorie, soluția propusă este eficientă, deoarece va utiliza doar un vector de 10 componente şi variabile simple de lucru. Pentru o descriere coerentă a metodei se acordă 1p., iar pentru justificarea eficienței, încă 1p. - în total **2p.**

DISCIPLINA INFORMATICĂ

Specializările matematică-informatică și matematică-informatică, intensiv informatică

I. STATUTUL DISCIPLINEI

În cadrul examenului de bacalaureat național, INFORMATICA are statutul de disciplină opțională, fiind susținută la proba E. d) în funcție de filieră, profil și specializare.

II. COMPETENȚE DE EVALUAT

- construirea algoritmilor corespunzători unor prelucrări elementare și reprezentarea lor în pseudocod și prin programe scrise în limbaj de programare (Pascal, C sau C++, la alegere);
- analiza rezolvării unei probleme prin urmărirea evoluției valorilor variabilelor prelucrate de algoritmul corespunzător;
- abstractizarea rezolvării prin construirea unor algoritmi echivalenți;
- identificarea și utilizarea tipurilor de date predefinite specifice unui limbaj de programare;
- definirea și utilizarea unor tipuri de date proprii;
- identificarea și utilizarea operatorilor predefiniți elementari;
- identificarea și utilizarea subprogramelor predefinite elementare;
- identificarea și utilizarea regulilor sintactice specifice limbajului de programare studiat;
- definirea și apelul unor subprograme proprii cu înțelegerea mecanismelor de transfer prin intermediul parametrilor;
- identificarea proprietăților unor structuri de date necesare în rezolvarea problemelor cu ajutorul calculatorului și utilizarea unor modele de memorare a acestora;
- organizarea datelor ce intervin în rezolvarea unei probleme utilizând structuri de date adecvate;
- organizarea etapelor de prelucrare ce formează un algoritm utilizând structuri de control și module de program;
- folosirea unor metode sistematice de rezolvare pentru probleme de generare;
- analiza unor algoritmi echivalenți de rezolvare a unei probleme în vederea alegerii algoritmului optim.

III. CONȚINUTURI

1. **Algoritmi**
   - 1.1. Noțiunea de algoritm, caracteristici
   - 1.2. Date, variabile, expresii, operații
   - 1.3. Structuri de bază: liniară, alternativă și repetitivă
   - 1.4. Descrierea algoritmilor, reprezentare în pseudocod
2. **Elementele de bază ale unui limbaj de programare (Pascal sau C, la alegere)**
   - 2.1. Vocabularul limbajului
   - 2.2. Constante. Identificatori
   - 2.3. Noțiunea de tip de dată. Operatori aritmetici, logici, relaționali
   - 2.4. Definirea tipurilor de date
   - 2.5. Variabile. Declararea variabilelor
   - 2.1. Structura programelor. Comentarii
   - 2.2. Expresii. Instrucțiunea de atribuire
   - 2.3. Citirea /scrierea datelor
   - 2.4. Structuri de control: instrucțiunea compusă, structuri alternative și repetitive

3. Subprograme predefinite
- 3.1. Subprograme. Mecanisme de transfer prin intermediul parametrilor
- 3.2. Proceduri și funcții predefinite

4. Tipuri structurate de date
- 4.1. Tipul tablou
- 4.2. Tipul șir de caractere: operatori, proceduri și funcții predefinite pentru: citire, afișare, concatenare, căutare, extragere, inserare, eliminare
- 4.3. Tipul înregistrare

5. Fișiere text
- 5.1. Fișiere text. Tipuri de acces
- 5.2. Proceduri și funcții predefinite pentru fișiere text

6. Algoritmi elementari
- 6.1. Probleme care operează asupra cifrelor unui număr
- 6.2. Divizibilitate. Numere prime. Algoritmul lui Euclid
- 6.3. Șirul lui Fibonacci
- 6.4. Determinare minim/maxim
- 6.5. Metode de ordonare: metoda bulelor, selecției
- 6.6. Interclasare
- 6.7. Metode de căutare: secvențială, binară
- 6.8. Analiza complexității unui algoritm considerând criteriile de eficiență *durata de executare* și *spațiu de memorie utilizat*

7. Subprograme definite de utilizator
- 7.1. Proceduri și funcții: declarare și apel, parametri formali și parametri efectivi, parametri transmiși prin valoare, parametri transmiși prin referință, variabile globale și variabile locale, domeniu de vizibilitate
- 7.2. Proiectarea modulară a rezolvării unei probleme

8. Recursivitate
- 8.1. Proceduri și funcții recursive

9. Metoda backtracking
- 9.1. Probleme de generare

10. Generarea elementelor combinatoriale
- 10.1. Permutări, aranjamente, combinări
- 10.2. Produs cartezian, submulțimi

11. Grafuri
- 11.1. Grafuri neorientate: terminologie (nod/vârf, muchie, adiacență, incidență, grad, lanț, lanț elementar, ciclu, ciclu elementar, lungime, subgraf, graf parțial), proprietăți (conex, componentă conexă, graf complet), metode de reprezentare în memorie (matrice de adiacență, liste de adiacență)
- 11.2. Arbori: terminologie (nod, muchie, rădăcină, descendent, descendent direct/fiu, ascendent, ascendent direct/părinte, frați, nod terminal, frunză), metode de reprezentare în memorie (matrice de adiacență, vector de „tați")

NOTĂ: Programa de examen este realizată în conformitate cu prevederile programelor școlare în vigoare pentru învățământul liceal. Variantele de subiecte pentru examenul național de bacalaureat evaluează competențele și conținuturile din prezenta programă, iar baremele de evaluare și de notare prevăd acordarea punctajelor pentru orice modalitate corectă de rezolvare a cerințelor. Variantele de subiecte pentru examenul național de bacalaureat se elaborează în baza prezentei programe și nu vizează conținutul unui manual anume.

DISCIPLINA INFORMATICĂ
Specializarea științe ale naturii

I. STATUTUL DISCIPLINEI

În cadrul examenului de bacalaureat național, INFORMATICA are statutul de disciplină opțională, fiind susținută la proba E. d) în funcție de filieră, profil și specializare.

II. COMPETENȚE DE EVALUAT

- construirea algoritmilor corespunzători unor prelucrări elementare și reprezentarea lor în pseudocod și prin programe scrise în limbaj de programare (Pascal, C sau C++, la alegere);
- analiza rezolvării unei probleme prin urmărirea evoluției valorilor variabilelor prelucrate de algoritmul corespunzător;
- abstractizarea rezolvării prin construirea unor algoritmi echivalenți;
- identificarea și utilizarea tipurilor de date predefinite specifice unui limbaj de programare;
- definirea și utilizarea unor tipuri de date proprii;
- identificarea și utilizarea operatorilor predefiniți elementari;
- identificarea și utilizarea subprogramelor predefinite elementare;
- identificarea și utilizarea regulilor sintactice specifice limbajului de programare studiat;
- identificarea proprietăților unor structuri de date necesare în rezolvarea problemelor cu ajutorul calculatorului și utilizarea unor modele de memorare a acestora;
- organizarea datelor ce intervin în rezolvarea unei probleme utilizând structuri de date adecvate;
- organizarea etapelor de prelucrare ce formează un algoritm utilizând structuri de control;
- analiza unor algoritmi echivalenți de rezolvare a unei probleme în vederea alegerii algoritmului optim.

III. CONȚINUTURI

1. **Algoritmi**
   - 1.1. Noțiunea de algoritm, caracteristici
   - 1.2. Date, variabile, expresii, operații
   - 1.3. Structuri de bază: liniară, alternativă și repetitivă
   - 1.4. Descrierea algoritmilor, reprezentare în pseudocod
2. **Elementele de bază ale unui limbaj de programare (Pascal sau C, la alegere)**
   - 2.1. Vocabularul limbajului
   - 2.2. Constante. Identificatori
   - 2.3. Noțiunea de tip de dată. Operatori aritmetici, logici, relaționali
   - 2.4. Definirea tipurilor de date
   - 2.5. Variabile. Declararea variabilelor
   - 2.6. Structura programelor. Comentarii
   - 2.7. Expresii. Instrucțiunea de atribuire
   - 2.8. Citirea/scrierea datelor
   - 2.9. Structuri de control: instrucțiunea compusă, structuri alternative și repetitive
3. **Subprograme predefinite**
   - 3.1. Subprograme. Mecanisme de transfer prin intermediul parametrilor
   - 3.2. Proceduri și funcții predefinite
4. **Tipuri structurate de date**
   - 4.1. Tipul tablou – tablouri unidimensionale

5. Fișiere text

- 5.1. Fișiere text. Tipuri de acces
- 5.2. Proceduri și funcții predefinite pentru fișiere text

6. Algoritmi elementari

- 6.1. Probleme care operează asupra cifrelor unui număr
- 6.2. Divizibilitate. Numere prime. Algoritmul lui Euclid
- 6.3. Șirul lui Fibonacci
- 6.4. Determinare minim/ maxim
- 6.5. Metode de ordonare: metoda bulelor, selecției
- 6.6. Interclasare
- 6.7. Metode de căutare: secvențială, binară
- 6.8. Analiza complexității unui algoritm considerând criteriile de eficiență *durata de executare* și *spațiu de memorie utilizat*

NOTĂ: Programa de examen este realizată în conformitate cu prevederile programelor școlare în vigoare pentru învățământul liceal. Variantele de subiecte pentru examenul național de bacalaureat evaluează competențele și conținuturile din prezenta programă, iar baremele de evaluare și de notare prevăd acordarea punctajelor pentru orice modalitate corectă de rezolvare a cerințelor. Variantele de subiecte pentru examenul național de bacalaureat se elaborează în baza prezentei programe și nu vizează conținutul unui manual anume.

DE CE NU PYTHON 3?

Cursul *online interactiv gratuit* **respectă programa școlară aprobată M.E.N.** și poate fi folosit în cadrul oricărui ciclu de învățământ ca o introducere în limbajul de programare **Python 3**. Vezi mai multe detalii pe [www.pythonisti.ro]:



Limbajul Python este folosit la scară largă în prezent, fiind foarte popular, simplu de învățat și de utilizat. Totul a pornit ca un hobby de Crăciun pentru **Guido van Rossum** în anul 1989 – fiind programator, a dorit să dezvolte un interpretor de cod pentru limbajul la care visa; simplu, intuitiv, open source, accesibil pe orice platformă. Deoarece urmărea cu drag serialul difuzat de BBC – **Monty Python's Flying Circus**, l-a numit *Python*:





De ce pitoni atunci?

Până la urmă, "**python**" semnifică în limba engleză *un piton*. Chiar dacă denumirea provine de la acel serial TV, priviți puțin logo-ul oficial al limbajului de programare:

Deci, precum vedem, avem doi pitoni, clar! :) Extensia oricărui fișier *Python* este "**py**". De asemenea, priviți culorile cu care este realizat genericul serialului...

De ce să învățați limbajul Python?

Cele mai importante avantaje sunt evidențiate mai jos:

• **facil de învățat / predat** – spre deosebire de alte limbaje de programare, Python se înțelege foarte repede, chiar de la început;

• **facil de folosit** – Python este simplu și minimalist, apropiat de pseudocod în limba engleză și permite ca atenția să fie focalizată pe găsirea soluției problemei și nu utilizarea limbajului în sine;

• **gratuit și open source** – este permisă redistribuirea limbajului, a codului sursă, precum și editarea / modificarea acestuia pentru a obține spre exemplu noi programe, deci poate fi și este îmbunătățit de comunitate în mod constant;

• **portabil** – limbajul Python poate fi utilizat pe o multitudine de platforme / sisteme de operare, precum: Windows, GNU/Linux, FreeBSD, Macintosh, Solaris, OS/2, PlayStation, etc.;

• **puternic orientat pe obiecte** – într-un mod foarte simplist, se pot crea cu succes aplicații folosind Programarea Orientată pe Obiecte (POO) - cam orice este un obiect, de aici și forța acestui limbaj de programare!

Python este *simplu* și *minimalist*, apropiat de pseudocod în limba engleză și permite ca atenția să fie focalizată pe *găsirea soluției problemei și nu utilizarea limbajului în sine*.

Poți învăța limbajul ***Python 3*** de pe orice dispozitiv mobile. Vezi mai multe pe:

[www.pythonisti.ro]

infobits Academy | [www.infobits.ro]
28+ de ani de educație în România